A Recurrent Convolutional Neural Network Approach for Sensorless Force Estimation in Robotic Surgery

Arturo Marban[a,c,*], Vignesh Srinivasan[c], Wojciech Samek[c], Josep Fernández[a], Alicia Casals[a,b]

[a] *Research Centre for Biomedical Engineering (CREB), Universitat Politècnica de Catalunya, 08034 Barcelona, Spain*
[b] *Institute for Bioengineering of Catalonia (IBEC), The Barcelona Institute of Science and Technology (BIST), 08028 Barcelona, Spain*
[c] *Machine Learning Group, Fraunhofer Heinrich Hertz Institute, 10587 Berlin, Germany*

Abstract

Providing force feedback as relevant information in current Robot-Assisted Minimally Invasive Surgery systems constitutes a technological challenge due to the constraints imposed by the surgical environment. In this context, force estimation techniques represent a potential solution, enabling to sense the interaction forces between the surgical instruments and soft-tissues. Specifically, if visual feedback is available for observing soft-tissues' deformation, this feedback can be used to estimate the forces applied to these tissues. To this end, a force estimation model, based on Convolutional Neural Networks and Long-Short Term Memory networks, is proposed in this work. This model is designed to process both, the spatiotemporal information present in video sequences and the temporal structure of tool data (the surgical tool-tip trajectory and its grasping status). A series of analyses are carried out to reveal the advantages of the proposal and the challenges that remain for real applications. This research work focuses on two surgical task scenarios, referred to as pushing and pulling tissue. For these two scenarios, different input data modalities and their effect on the force estimation quality are investigated. These input data modalities are tool data, video sequences and a combination of both. The results suggest that the force estimation quality is better when both, the tool data and video sequences, are processed by the neural network model. Moreover, this study reveals the need for a loss function, designed to promote the modeling of smooth and sharp details found in force signals. Finally, the results show that the modeling of forces due to pulling tasks is more challenging than for the simplest pushing actions.

Keywords: Robotic Surgery, Force Estimation, Convolutional Neural Networks, LSTM Networks.

1. Introduction

Traditional open surgery, characterized by long incisions, has been improved by minimally invasive surgery, which uses long instruments inserted into the body through small incisions. An endoscopic camera provides visual feedback of the target scenario, and two or more surgical instruments allow the surgeon to interact with tissues and organs. Minimally invasive surgery has been extended and enhanced in capabilities by robotic teleoperated systems with a master-slave configuration, resulting in a new procedure known as Robotic Assisted Minimally Invasive Surgery (RAMIS) [1][2].

RAMIS provides surgeons with augmented capabilities, such as fine and dexterous movements, proper hand-eye coordination, hand tremor suppression and high-quality visualization of the surgical scenario [2]. Nonetheless, the integration of force feedback as relevant information in these systems still remains an open problem [3][4]. Force feedback has proven to be beneficial in teleoperated surgery since it is associated with the control of interaction forces and thus, its use can result in less intraoperative tissue damage produced by the application of excessive forces. Force feedback also helps to improve the proper execution of surgical tasks, such as grasping or suturing, in which the application of excessive or insufficient forces can produce damage or malfunctions. Furthermore, force feedback can provide information of tissue stiffness and shape. Therefore, it can help to detect abnormalities, such as tumors or calcified arteries [5].

The main difficulty in providing RAMIS systems with force feedback relies on measuring interaction forces between surgical instruments and tissues. This problem can be addressed by two approaches: direct force sensing and sensorless force estimation. In direct force sensing, the measurement of forces is carried out with a sensor located at, or close to, the point of interaction between tool and tissue. Although it represents the most intuitive solution, many constraints, such as biocompatibility, sterilization, miniaturization, and cost [6], limit the design of such force sensors. The need of miniaturization has been addressed in different works such as [7], where a laparoscopic instru-

*Corresponding author
Email addresses: arturo.marban@hhi-extern.fraunhofer.de (Arturo Marban), vignesh.srinivasan@hhi.fraunhofer.de (Vignesh Srinivasan), wojciech.samek@hhi.fraunhofer.de (Wojciech Samek), josep.fernandez@upc.edu (Josep Fernández), alicia.casals@upc.edu (Alicia Casals)

ment with force sensing capability is described. However, its clinical validation has not been proven yet, since it was only tested in an open platform for surgical robotics research, called Raven-II [8]. In contrast, force estimation allows the removal of any electronic device from the instrument in contact with the patient. Therefore, the interaction forces have to be estimated from the available sources of information, which may result in inaccurate measures. Due to the aforesaid reasons, sensorless force estimation represents a potential solution for the practical implementation of force perception systems in RAMIS.

Sensorless force estimation can be implemented through control-based or vision-based approaches. In the control-based approach, interaction forces are estimated using observers and models of the surgical tool, and by processing available information from the motor units (i.e. angular position/velocity, current consumption, and torque). In this regard, some relevant works are focused on estimating the surgical instrument grasping force, as described in [9] and [10]. In contrast, the vision-based approach consists in estimating forces mainly from video sequences (monocular or stereo), therefore, in this work it is referred to as Vision-Based Force Sensing (VBFS). In VBFS, the uncertainty of the force estimates is reduced by having access to surgical tool data, such as tool-tip trajectory, its velocity, and grasper status. Although there are fewer works in the literature related to VBFS, if developed properly, it has potential to restore force feedback in robotic surgery. VBFS avoids the need for accurate modeling of the surgical instrument or slave-robot manipulator, as required by most control-based approaches.

In the next section, deep neural networks are introduced as effective models applied in the processing of video sequences (Section 1.1). Subsequently, the concept of VBFS is defined and different works reported in the literature are described (Section 1.2). Finally, the proposed approach for estimating forces in robotic surgery is presented and the contributions of this research work are listed (Section 1.3).

1.1. Deep Neural Networks for Processing Video Sequences

In recent years, Convolutional Neural Networks (CNN) have shone light in tasks related to the processing of images. These models hold the state of the art results in the task of image classification. In this context, some of the most representative CNN architectures that have been proposed are AlexNet [11], VGG16 [12], RESNET [13], and Inception [14]. A powerful property of CNNs is transfer learning [15]. That is, given a CNN trained in a base dataset and task, the learned features can be transferred to another CNN, to be trained in a different dataset and task. For instance, [16] shows that a pre-trained Inception model in the ImageNet dataset [17] (designed to classify natural images), can be used in the classification of images describing skin cancer lesions. This task was accomplished by fine-tuning the Inception model in a dataset of clinical images labeled with the corresponding skin lesions. The learned features by a pre-trained CNN can also be helpful in the processing of data with a temporal constraint, as in the application of video classification [18]. In the present work, the use of pre-trained models (i.e. the VGG16 network) and the concept of transfer learning are exploited in the force estimation task, as detailed later in Section 4.

In the processing of sequences of data with long-term temporal dependencies, Long-Short Term Memory (LSTM) networks [19] have excelled, providing state of the art results in applications such as language modeling and translation, speech synthesis, and analysis of audio and video data [20][21][22]. In particular, the LSTM network with coupled input-forget gates, suggested by [21] as a less computational expensive model than the vanilla LSTM network [23], was found suitable for the force estimation task, as discussed later in Section 5.3.

Deep neural networks composed of CNNs and LSTM networks have been investigated in different domains where the input data has a spatiotemporal structure, as in video sequences. The CNN addresses the processing of spatial information, while the LSTM network the processing of temporal information. This neural network architecture has been applied in action recognition with visual attention [24], video activity recognition and image captioning [25], video content description [26], and learning physical interaction through video prediction [27], among others. A particular domain of interest is related to the estimation of time-varying signals from video sequences in the context of a regression framework. In this regard, [28] proposed a technique to estimate sound from silent video sequences through a neural network consisting of a CNN and LSTM networks. This neural network was trained using a video dataset, describing interactions of a wooden stick with different objects and materials with added audio recordings. In another application, [29] developed a technique to estimate continuous pain intensity from video sequences of facial expressions. This technique is based on a CNN with added recurrent connections in its layers.

1.2. Vision-Based Force Sensing

The Vision-Based Force Sensing (VBFS) concept relies on a simple observation, that is, soft bodies made of biological (i.e. tissue) or artificial (i.e. silicone) materials deform under an applied load. Therefore, if the deformation of soft bodies (i.e. biological tissues) is available from visual feedback (i.e. video sequences), this feedback can be used to estimate the forces applied on these objects, [30][31]. VBFS methods are developed to estimate forces in 2D or 3D scenarios. In the first case, a force applied to a soft body results in a deformed contour, while in the second case, it produces a deformed surface.

Notable works, such as [31] and [32], developed the concept of VBFS in 2D scenarios using neural networks. This approach circumvents the explicit modeling of complex mechanical properties attributed to some materials (i.e. biological cells). In [31], VBFS is applied to estimate forces in objects that exhibit both linear (a microgripper) and

non-linear (a rubber torus) mechanical properties. This method relies on a deformable template matching algorithm to describe the object's contour deformation and a fully-connected neural network that models the object's mechanical properties. The micromanipulation of cells with a spherical shape has been addressed in [32]. In this work, a method is developed to estimate force during microinjection of zebrafish embryos. This method relies on active contours and conic fitting algorithms to model the cell's contour deformation. Then, a fully-connected neural network learns the non-linear relationship between deformation and force.

The estimation of interaction forces between tools and tissues becomes more realistic when tissue deformation is processed in 3D space, that is, by taking into account depth information. To this end, a stereo vision system is used to recover such information. Minimally invasive surgical procedures are complex, however, they can be interpreted as the composition of different elementary surgical tasks [33]. One of such tasks, referred to as pushing tissue (pressing the end of the endoscopic tools against soft-tissue), represents a common practice in minimally invasive surgery [34]. This surgical task is studied in the context of VBFS due to its simplicity.

Force estimation techniques that rely on a stereo vision system are reported in [34], [35], [36], [37] and [38]. In [34], the forces developed in a rubber membrane are studied. Its deformation was recovered by tracking nodal displacements and a finite element method was used to model the mechanical relationship between deformation and force. VBFS applied to neurosurgery was investigated in [35]. In this work, soft-tissue surface deformation is computed using a depth map extracted from stereo-endoscopic images. Thereafter, this information is processed by a surface mesh (based on spring-damper models) to render force as output. Another approach in the context of neurosurgery has been investigated in [36]. The authors of this work developed a method based on quasi-dense stereo correspondence to recover surface deformation from stereo video sequences. Afterward, force is estimated from the surgical tool displacement (which is extracted from the deformation data), using a 2nd order polynomial model. In recent years, models based on neural networks have been investigated. In this regard, [37] proposed a method consisting in a 3D lattice and a recurrent neural network. The 3D lattice models the complex deformation of soft-tissues. The recurrent neural network was designed to estimate force by processing the information provided by this lattice in addition to the surgical tool motion. A subsequent notable work by the same author is presented in [38]. In this work, the recurrent neural network described in [37] is improved by designing a model based on the LSTM network architecture, achieving high accuracy in the estimation of forces (in 3D space). Monocular force estimation represents a more challenging approach. In this regard, [39] developed a technique to estimate forces from monocular video sequences using a real lamb liver as experimental material. This method relies on a virtual template to model soft-tissue surface deformation, however, it assumes that soft-tissue surface behaves as a smooth function with local deformation. Then, a stress-strain bio-mechanical model defines the relationship between force and penetration depth caused by the surgical tool.

From the literature review a series of conclusions are drawn. First, most of the existing methods recover tissue deformation using a stereo vision system ([34]-[38]). They rely on a deformation model which is created based on 3D geometries such as a mesh or lattice (i.e. [35] and [38]), or stereo-correspondences (i.e. [36]). Second, the estimation of forces has been studied only for pushing tasks. Other surgical tasks that result in complex interactions, such as pulling or grasping tissue, have not been addressed yet. Third, recurrent neural network architectures have been studied in [37] and [38], performing a mapping from soft-tissue deformation and tool data to interaction force. From these two works, only [38] describes the use of a deep neural network, specifically a LSTM network. Fourth, CNNs, which excel in tasks related to processing spatial information present in images or video sequences (e.g., [11, 24, 40]) have not been explored in the processing of visual information available from RAMIS systems. Fifth, monocular force estimation was only addressed in [39]. Nonetheless, this method relies on feature detection and matching algorithms that are not robust to specularities produced by reflection of light on the tissue surface. Therefore, feature points had to be detected and matched manually during the reported experiments. Furthermore, the force was estimated only for the loading cycle (when the tool is incrementally deforming the tissue, before reaching the peak force), and for one component (F_z). Finally, due to the complexity of data acquisition (i.e. video sequences, tool data and force sensing) in a real surgical scenario, most methods ([34]-[38]) are implemented and validated on experimental platforms using organs made of artificial tissues (i.e. silicone). Only [39] describes experiments on a real lamb liver.

The literature review shows that an approach based on deep neural networks, specifically, CNN and LSTM networks, has not been investigated for VBFS in robotic surgery. Its advantages and downsides will reveal new research directions to design a better force estimation model that learns from data. In particular, transfer learning techniques (i.e. using a pre-trained CNN on the ImageNet dataset) have not been explored for VBFS in the context of robotic surgery. They can be useful to encode complex phenomena (i.e. tool-tissue interactions) in a low-dimensional feature vector representation learned from high-dimensional data, such as video sequences. This feature vector representation is easier to model by an LSTM network.

1.3. Recurrent Convolutional Neural Network Approach

In the present work, a Recurrent Convolutional Neural Network (RCNN) architecture, based on CNN and

LSTM networks, is proposed for VBFS in RAMIS. It estimates a 6-dimensional vector of forces and torques (in the 3D space) at every time instant, by processing monocular video sequences and tool data.

The focus of this research work is on the estimation of interaction forces in two surgical tasks, pushing (pressing the tool against a tissue) and pulling a tissue (which requires grasping). This surgical task decomposition was motivated by the discrete model presented in [33]. In that work, the complexity of minimally invasive surgical procedures is modeled taking into account a set of fundamental tasks, among them, pushing and pulling a tissue. Moreover, different input data modalities and their effect on the force estimation quality are investigated. These input data modalities are: (i) the tool data represented by the tool-tip trajectory (in 3D space) and its grasping status (opened/closed), (ii) video sequences, and (iii) a combination of both. Finally, to facilitate the modeling of smooth and sharp details found in the estimated force and torque signals, the RCNN is optimized with a loss function designed with the Root Mean Squared Error (RMSE) and Gradient Different Loss (GDL), respectively. The GDL has been investigated in the prediction of future frames from video sequences as discussed in [41], enabling a deep neural network to render sharp images, avoiding blurred pixels. Nonetheless, this concept has neither been extended nor studied for the prediction of time-varying signals.

Although models based on CNN and LSTM networks have been investigated in different domains (as discussed in Section 1.1), their application to the force estimation task comes with its own challenges. Therefore, two important goals of this research work are: (i) to reveal the advantages and downsides of a force estimation model based on deep neural networks, and (ii) define future research directions for its implementation on real scenarios. To this end, the following contributions are made:

- A RCNN model is proposed for the estimation of interaction forces between tool and tissue relying on a single camera. This method has potential applications in scenarios where a stereo vision system is unavailable, and consequently, depth information.
- The effectiveness of applying transfer learning techniques is investigated with the objective of finding a compact feature vector representation for every video frame. For this purpose, the pre-trained VGG16 network in the ImageNet dataset is used. This approach allows encoding complex phenomena described in video sequences, such as the deformation of tissues and specular reflections, in a feature vector representation automatically learned from data. This representation is easier to process by a model that learns sequences of data, such as an LSTM network.
- A loss function designed with the RMSE and GDL is investigated to facilitate the modeling of smooth and sharp details found in force/torque signals. This loss function composition provides more accurate force estimations than considering only RMSE during the RCNN optimization.
- Video pre-processing techniques, specifically mean frame removal and space-time transformations, discussed in [42] and [28] respectively, were studied to ease the learning process of the RCNN. Mean frame removal was found useful to discard those regions in video sequences which do not contribute to the learning process, such as the static background. The space-time transformation, allows emphasizing motion produced by tool-tissue interactions, in a new image representation created from three consecutive frames.

The next sections are organized as follows. Section 2 defines the problem statement. Section 3 describes the dataset acquisition using an experimental robotic platform, and the pre-processing operations applied to this data. Section 4 details the proposed RCNN architecture for force estimation. Section 5 presents the experiments, providing details related to the two stage RCNN optimization, and describes how the robustness of the RCNN model was evaluated. Section 6 discusses the results of the experiments and analyses the quality of the estimated force signals with different metrics. Finally, Section 7 presents the conclusions and future work.

2. Problem Statement

Given sequences of video frames $X_t^{video} \in \Re^{h \times w \times c}$ (h, w and c stand for image height, width and number of channels, respectively) and tool data $X_t^{tool} \in \Re^8$, the objective is to find a non-linear model $\mathcal{F}(.)$ with parameters $\mathcal{W}$, that maps X_t^{video} and X_t^{tool} to a sequence of estimated forces $\widehat{Y}_t \in \Re^6$ at each time instant t, as expressed in Equation (1).

$$\widehat{Y}_t = \mathcal{F}(X_t^{tool}, X_t^{video}; \mathcal{W}) \tag{1}$$

The elements of the input vector X_t^{tool} are shown in Equation (2), where $P_t^{tool} = [x_t, y_t, z_t]$ is a vector describing the tool-tip trajectory in the 3D space, $\Lambda_t^{tool} = [u_t, v_t, w_t]$ is an unitary vector that defines the tool orientation in 3D space (coincident with the tool-axis direction), θ_t is the angle of rotation around this axis, and s_t is the tool grasper status, defined in Equation (3). The tool-tip trajectory (P_t^{tool}) and its orientation (defined by Λ_t^{tool} and θ_t) are illustrated in Fig. 1. The elements of the output vector $\widehat{Y}_t$ (shown on the left of Equation (1)) are the estimated forces, $\hat{F}_t = [\hat{f}_t^x, \hat{f}_t^y, \hat{f}_t^z]$, and torques, $\hat{T}_t = [\hat{\tau}_t^x, \hat{\tau}_t^y, \hat{\tau}_t^z]$, in the 3D space. Thus, $\widehat{Y}_t = [\hat{F}_t, \hat{T}_t]' = [\hat{f}_t^x, \hat{f}_t^y, \hat{f}_t^z, \hat{\tau}_t^x, \hat{\tau}_t^y, \hat{\tau}_t^z]'$.

$$X_t^{tool} = [P_t^{tool}, \Lambda_t^{tool}, \theta_t, s_t]' \tag{2}$$

$$s_t = \begin{cases} 1 & \text{If the grasper is open.} \\ 0 & \text{If the grasper is closed.} \end{cases} \tag{3}$$

In the present work, $\mathcal{F}(.)$ is learnt from data by using a deep neural network. Therefore, given a rich dataset $\mathcal{D}$

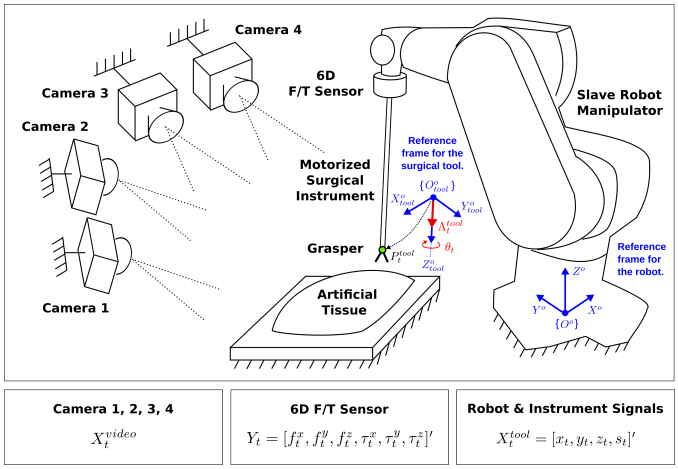


Figure 1: Diagram of the experimental setup used to create the dataset. In the bottom, the three blocks relate devices/sensors to the recorded data (in vector form). $\{O^o\}$ is the reference frame assigned to the robot with axes, X^o, Y^o, and Z^o, while $\{O^o_{tool}\}$ is the reference frame of the surgical tool-tip with respect to the robot with axes, X^o_{tool}, Y^o_{tool}, and Z^o_{tool}. The origin of $\{O^o_{tool}\}$ is located at the tool-tip, and its Z^o_{tool} axis is aligned with the tool shaft. Therefore, the origin of $\{O^o_{tool}\}$ describes the tool-tip trajectory at each time instant t, $P_t^{tool} = [x_t, y_t, z_t]$. The tool orientation is defined by the unitary vector $\Lambda_t^{tool} = [u_t, v_t, w_t]$ and the scalar θ_t. The vector Λ_t^{tool} has the same direction as the Z^o_{tool} axis.

consisting of video sequences X_t^{video}, tool data X_t^{tool} and ground-truth interaction forces Y_t, the goal is to find the parameters $\mathcal{W}$ that satisfy Equation (1) in the context of an optimization framework. A causal constraint is enforced, that is, a estimated force vector $\widehat{Y}_t$ at the current time step, is computed by processing samples from X_t^{video} and X_t^{tool} at the current and previous time steps (i.e. t, $t-1$, $t-2$, $t-3$, ...). In the reported methodology and experiments, the tool orientation remained fixed, therefore, $X_t^{tool} = [P_t^{tool}, s_t] = [x_t, y_t, z_t, s_t]' \in \Re^4$. Nonetheless, in the general case, the full vector $X_t^{tool} \in \Re^8$ should be considered.

3. Dataset Acquisition & Pre-processing

Due to the lack of public datasets related to the application of VBFS in RAMIS, an experimental platform was designed to evaluate the proposed approach, as depicted in Fig. 1. This platform was used to record video sequences, tool data, and ground-truth interaction forces:

- **Video Sequences.** A collection of 44 video sequences, totaling 4.31 hours, were recorded using 4 digital cameras (DFK 72BUC02) with the objective to provide rich visual information from different perspectives. The four cameras were synchronized and the video sequences were recorded with a resolution of 480×640 pixels at 50 frames per second, in RGB color space. The target scenario consists in a motorized surgical instrument with grasping capability, mounted on a slave robot manipulator (Stäubli RX60B) that interacts with a digestive apparatus made of artificial tissue (Silicone-Smooth On ECOFLEX 0030). A sample of frames captured by the 4 cameras illustrates the aforesaid scenario in Fig. 2. They show specularities and highlights rendered on the artificial tissue surface, a phenomenon that is present in real minimally invasive surgery scenarios.

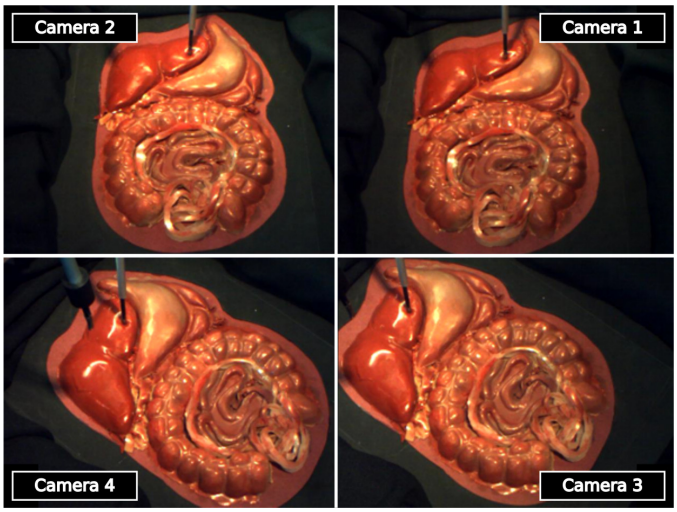


Figure 2: A sample of video frames recorded by the four synchronized cameras. The tool is performing a pushing task over the artificial organs (digestive apparatus).

- **Tool Data.** The tool-tip trajectory in the 3D space ($P_t^{tool} = [x_t, y_t, z_t]$) and the tool grasping status (s_t) were provided, at each time instant, by the slave robot manipulator and the motorized surgical instrument, respectively.
- **Ground-Truth Force.** The interaction forces and torques between the surgical instrument tip and artificial tissue were acquired by a 6D force/torque sensor (ATI Gamma SI-32-2.5) with its z axis aligned with the surgical instrument shaft. The measured forces lie in the range +2.5/-10 N and the torques in ±5 Nm, which are consistent with those values reported in a real scenario [43].

Thereafter, a series of pre-processing operations were applied to the tool data, ground-truth interaction force and video frames. The pre-processing of the tool-tip trajectory $P_t^{tool} = [x_t, y_t, z_t]$, was carried out by removing the mean and subsequently scaling its amplitude to the range ±1. The grasper status s_t does not need any processing. The ground-truth interaction forces, Y_t, were compensated with an offset and scaled to the range -0.7/+0.5. In this representation, the force components are dimensionless, with a mean close to zero and similar variances. This normalization procedure is suggested in [44]. Additional processing steps, such as time shifting and resampling, were applied to both, tool data and ground-truth forces, to synchronize them with the video frames. Moreover, a low-pass filter was used to remove the noise from the ground-truth force data.

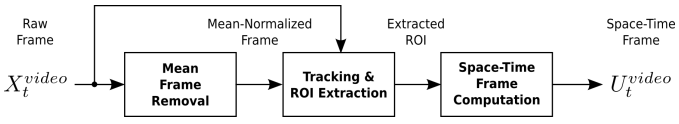
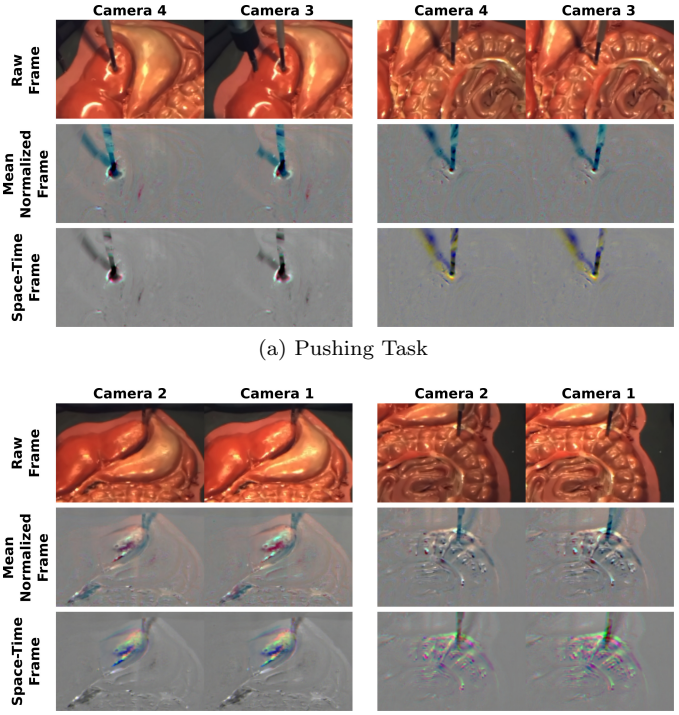


Figure 3: Block diagram of the pre-processing steps applied to video frames.

Video frames required more elaborated pre-processing steps, which can be summarized in the block diagram shown in Fig. 3, where X_t^{video} and U_t^{video} represent the raw and pre-processed video frames, respectively. Each operation in the block diagram was implemented using OpenCV [45] and is described as follows:

1. **Mean Frame Removal**. A mean frame was computed for every video sequence by averaging all the raw frames (with equal contribution). Subsequently, a subtraction operation was performed over the RGB channels, by removing the corresponding mean frame from all the raw frames in the corresponding video sequence. During this process, the pixel values were scaled properly, to conserve negative values. In [42], this method was shown to reduce over-fitting of CNNs due to static background present in video sequences.
2. **Tracking of Regions of Interest**. To provide meaningful visual information to the proposed network, a region of interest of dimensions 200×300 pixels, corresponding to the area of interaction between tool and tissue, was tracked and extracted from every mean-normalized frame (480×640 pixels). This operation was carried out by processing mean-normalized and raw frames. The result is a mask of foreground pixels describing image regions where tool-tip motion is present. For this purpose, each RGB frame was filtered with a non-local means denoising algorithm [46] and converted to grayscale. Afterward, a mask of foreground pixels was computed based on image differences between the current and a finite sequence of past frames (including a frame of the static scenario), followed by denoising (with a normalized box filter) and thresholding (to get the actual mask) operations. Finally, this mask was refined with morphological operations (i.e. erosion and dilation).
3. **Space-Time Frame Transformation**. This transformation, described in [28], is applied over the extracted regions of interest with the objective to model tool motion and tissue deformation. It represents an alternative method to the optical flow, which is computationally more expensive. A space-time frame is defined by the previous, current and next RGB frames, each one converted to grayscale. During the experiments, this operation was carried out by concatenating these three frames only every 15 samples. This undersampling is due to the high frame rate of the cameras and the slow motion of the surgical tool.

A comparison between regions of interest extracted from the raw, mean-normalized and space-time frames is presented in Fig. 4, for each surgical task. The last row of Fig. 4a and Fig. 4b shows that both, tool motion and tissue deformation are emphasized in the space-time domain, and specular reflections are partially suppressed.



(a) Pushing Task

(b) Pulling Task

Figure 4: A sample of raw video frames after the mean frame has been removed and the space-time transformation has been applied, for each surgical task.

4. Force Estimation Model

The intuition behind the design of the force estimation model is guided by the structure of the input and output data to be processed. Video sequences can be interpreted as data with a spatiotemporal structure. On the other hand, tool data and interaction forces, represent sequences data with only a temporal structure. Therefore, the force estimation model should be designed as a function that maps an input sequence (i.e. video sequences and tool data) to an output sequence (interaction forces), while preserving the structure of data. For this aim, a Recurrent Convolutional Neural Network (RCNN) is proposed to carry out the force estimation task. It consists of a Convolutional Neural Network (CNN) serially connected with a Long-Short Term Memory (LSTM) network. The RCNN is depicted in Fig. 5. This illustration shows the flow of data from the input to the output in four stages, and each neural network is optimized separately (as described in the second and fourth stages):

- First, pre-processing operations are applied to the raw video sequences (in RGB color space with a resolution

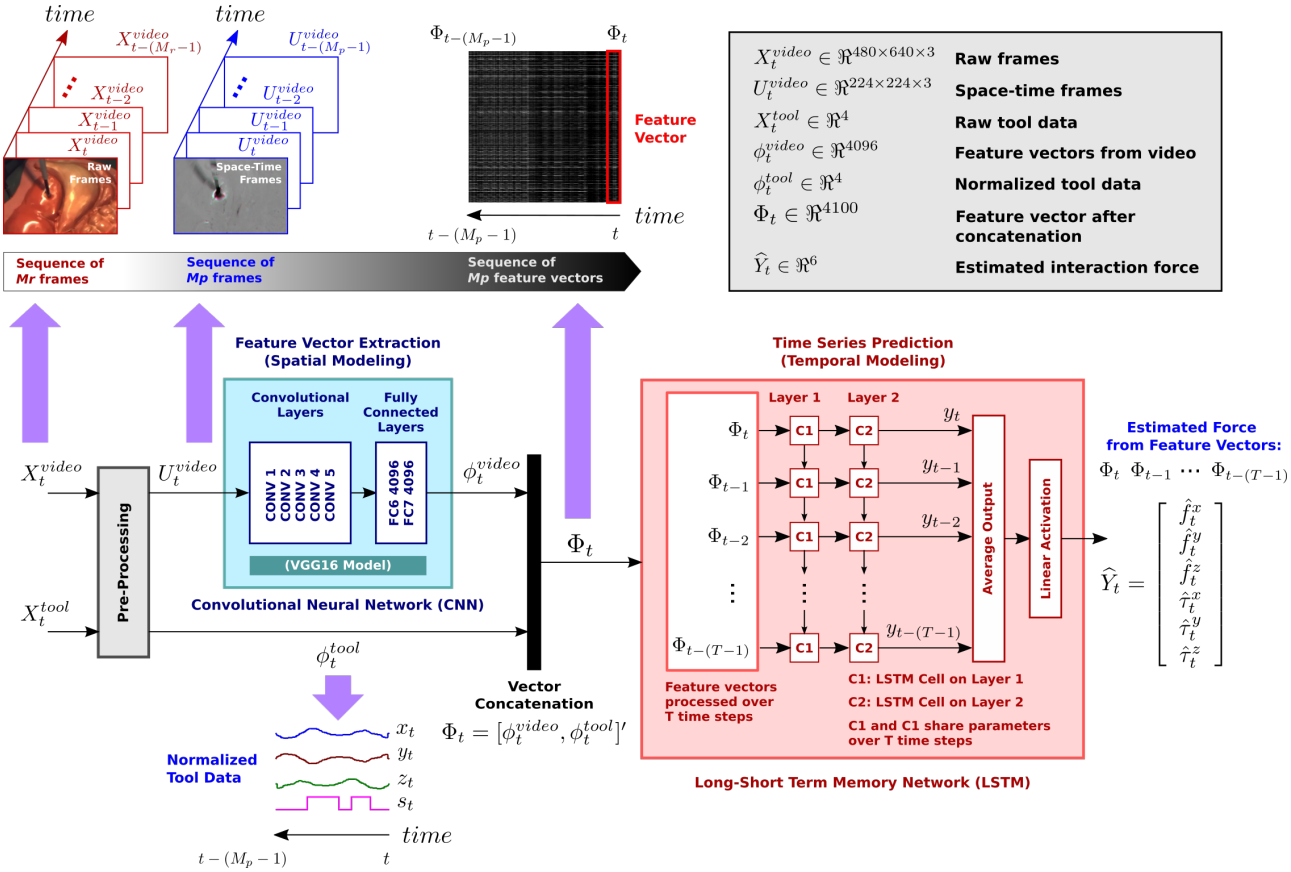


Figure 5: The RCNN architecture consists in a CNN serially connected with an LSTM network. First, pre-processing operations are applied to the input data consisting of raw video sequences (X_t^{video}) and tool data (X_t^{tool}). Therefore, a sequence of raw data (X_t^{video} and X_t^{tool}) of size M_r is transformed into a new sequence of pre-processed data (U_t^{video} and ϕ_t^{tool}, respectively) of size M_p, where $M_p < M_r$. The size difference of these two sequences results from the space-time transformation applied to raw video frames, which is computed by concatenating three consecutive (grayscale) frames spaced in time (in the experiments this spacing correspond to 15 frames). Subsequently, the CNN extracts feature vectors (ϕ_t^{video}) from the pre-processed input video sequence (U_t^{video}). Afterwards, these feature vectors (ϕ_t^{video}) and the normalized tool data (ϕ_t^{tool}) are concatenated, resulting in a new feature vector (Φ_t). Finally, these new feature vectors (Φ_t) are fed into the LSTM network, which models their temporal structure to render the estimated force as output ($\widehat{Y}_t$).

of 480 × 640 pixels), $X_t^{video} \in \Re^{480\times640\times3}$, and tool data, $X_t^{tool} \in \Re^4$, resulting in the space-time frames (in RGB color space with a resolution of 224 × 224 pixels), $U_t^{video} \in \Re^{224\times224\times3}$, and the normalized tool data, $\phi_t^{tool} \in \Re^4$, respectively.

- Second, the modeling of the spatial information present in video sequences is carried out by the CNN, specifically, the pre-trained VGG16 network model [12] (shown in Fig. 5 as the block in blue color). In the training stage, this neural network is optimized for a regression task on the dataset. The input and output data consist of space-time frames, U_t^{video}, and ground-truth interaction forces, $Y_t \in \Re^6$, respectively. Subsequently, in the inference stage, the VGG16 network is used as a feature extractor. It computes a feature vector representation, $\phi_t^{video} \in \Re^{4096}$, which encodes high-level abstractions of the input data, U_t^{video}. The VGG16 network and the feature vector extraction process are detailed in Section 4.1.

- Third, the information present in tool data and video sequences is encoded in a single feature vector representation, Φ_t. For this purpose, the feature vectors ϕ_t^{tool} (the normalized tool data) and ϕ_t^{video} (computed by the VGG16 network), are concatenated, resulting in $\Phi_t = [\phi_t^{video}, \phi_t^{tool}]' \in \Re^{4100}$.

- Fourth, the temporal information present in the new feature vector representation, Φ_t, is modeled by the LSTM network over T time steps (shown in Fig. 5 as the block in red color). In the training stage, the LSTM network is optimized for a regression task, by taking a sequence of feature vectors Φ_t as input (at the current and previous time steps, i.e. Φ_t, Φ_{t-1}, Φ_{t-2}, ..., $\Phi_{t-(T-1)}$), and a sequence of ground-truth interaction forces $Y_t \in \Re^6$ as output (at the current time step t). Thereafter, in the inference stage, the LSTM network processes a sequence of feature vectors Φ_t (i.e. Φ_t, Φ_{t-1}, Φ_{t-2}, ..., $\Phi_{t-(T-1)}$), to estimate a single force vector $\widehat{Y}_t \in \Re^6$, at the current time instant t.

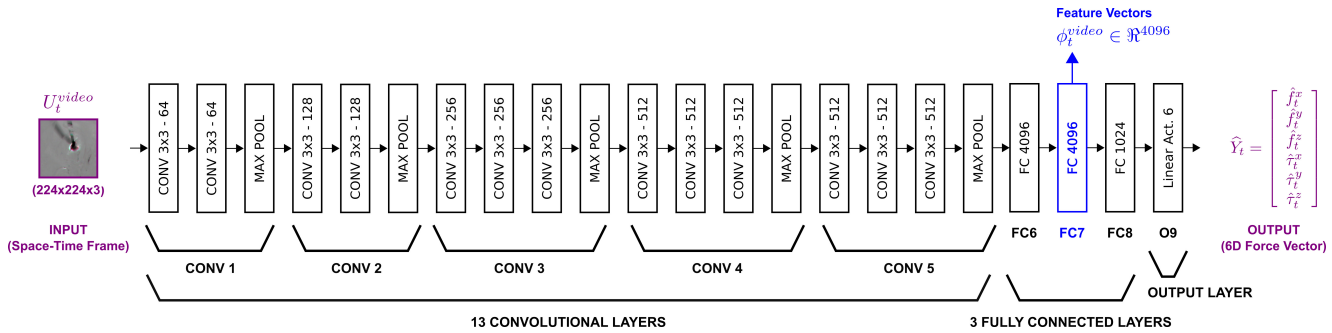


Figure 6: VGG16 network [12] used for fine-tuning and feature vector extraction. It consists of 13 convolutional (kernel size of 3×3) and 3 fully-connected layers. In this illustration, the convolutional layers are grouped into CONV 1, ..., CONV 5. The fully connected layers are referred to as FC6, FC7, and FC8. The rectified linear unit is used as activation function in all layers except the output layer, O9, which is densely connected with a linear activation. The number of output feature maps for each convolutional layer and the size of each fully connected layer are indicated with the last number inside the corresponding layer. At test time, the feature vectors $\phi_t^{video} \in \Re^{4096}$, are extracted from the layer FC7 (shown in blue color).

4.1. Feature Vector Extraction from Video Sequences

For the task of feature vector extraction from video sequences, the pre-trained VGG16 network was fine-tuned on the dataset. Specifically, in this process, the VGG16 network computes a force vector as output conditioned on an input video frame, while the network's parameters, in all layers, are adjusted in the context of an optimization framework. During the fine-tuning process, generic features (i.e. computed in the first and second layers) are less prone to change, while specific features (i.e. computed towards the last layer) will be adjusted according to the force estimation dataset. The VGG16 network, shown in Fig. 5 as the block in blue color, is detailed in Fig. 6. To match the neural network output size with that of the force vectors, the softmax layer of dimension 1000 (found in the original VGG16 network), was replaced by a densely connected layer of dimension 6 (with linear activation). Thus, only these parameters were optimized from scratch. The space-time frames (U_t^{video}) were resized preserving their aspect ratio (by centered cropping and resampling operations), from 200×300 to 224×224 pixels (matching the network's input size). After the fine-tuning process is completed, the feature vectors ϕ_t^{video}, are extracted from the fully-connected layer FC7 (shown in Fig. 6 in blue color).

4.2. Loss Function Design

The loss function has an important impact in the design of deep neural networks applied to regression tasks. This impact is also extended to the design of regression models based on CNNs. For instance, human pose estimation was studied in [42] with a CNN optimized with the standard $L2$ loss function (sensitive to outliers) to penalize the distance between predicted and ground-truth upper-body joint positions. The same application was investigated in [47], by minimizing Tukey's bi-weight function to achieve robustness against outliers. Recently, [41] proposed a method for predicting future images from a video sequence by the minimization of a loss function that takes into account the Gradient Different Loss (GDL). This method allows overcoming the prediction of blurry images when only the mean squared error is considered in the loss function. In the present work, the GDL has been extended to the estimation of time-varying force signals. Therefore, each network (CNN and LSTM), that defines the proposed RCNN architecture was optimized separately with a loss function composed of the Root Mean Squared Error (RMSE), and the GDL. The RMSE penalizes the distance between estimated and ground-truth 6D force vectors, while the GDL the distance between their gradients. Intuitively, the RMSE and GDL ease the modeling of smooth and sharp details found in force/torque signals, respectively.

The loss function discussed above, denoted as $\mathcal{L} \in \Re$, is mathematically expressed in Equation (4), where $\alpha \in [0, 1]$ represents a trade-off between the RMSE ($\mathcal{L}_{RMSE} \in \Re$) and GDL ($\mathcal{L}_{GDL} \in \Re$). The RMSE, expressed in Equation (5), computes the distance between the ground-truth $Y_i^{(j)} \in \Re$ and the estimated $\widehat{Y}_i^{(j)} \in \Re$ force components, where i indexes the samples in the dataset $\mathcal{D}$ and j the N force components. In this equation, $\rho(x_i) \in \Re$ is a function applied to the scalar $x_i \in \Re$, which is computed for the i-th sample in the dataset. The parameters described for the RMSE are also found in the GDL expressed in Equation (6).

$$\mathcal{L} = \alpha\, \mathcal{L}_{RMSE} + (1 - \alpha)\, \mathcal{L}_{GDL} \tag{4}$$

$$\mathcal{L}_{RMSE} = \sum_{i=1}^{|\mathcal{D}|} \rho(x_i),\ x_i = \sqrt{\frac{1}{N} \sum_{j=1}^{N} (Y_i^{(j)} - \widehat{Y}_i^{(j)})^2} \tag{5}$$

$$\mathcal{L}_{GDL} = \sum_{i=1}^{|\mathcal{D}|} \rho(x_i),\ x_i = \sum_{j=1}^{N} \left| |Y_i^{(j)} - Y_{i-1}^{(j)}| - |\widehat{Y}_i^{(j)} - \widehat{Y}_{i-1}^{(j)}| \right| \tag{6}$$

As mentioned in the beginning of this section, the RCNN optimization consists in two stages. In the first stage, the VGG16 network (shown in Fig. 6) is fine-tuned with a loss function defined in Equations (4)-(6). This neural network $\mathcal{F}_1$ with parameters $\mathcal{W}_1$, is represented by Equation (7), where $\widehat{Y}_i \in \Re^N$ stands for the estimated force vector, given

as input the i-th space-time frame, U_i^{video}. In the subsequent stage, the LSTM network $\mathcal{F}_2$ with parameters $\mathcal{W}_2$ shared across T time steps, is trained using the same loss function. This neural network is expressed in Equation (8). It outputs $\widehat{Y}_i \in \Re^N$, that is, the estimated force vector at the time instant i, given as input a sequence of T feature vectors Φ_d, at time steps $d = i, i-1, i-2, ..., i-(T-1)$, (see the LSTM network depicted in Fig. 5).

$$\widehat{Y}_i = \mathcal{F}_1(U_i^{video}; \mathcal{W}_1) \tag{7}$$

$$\widehat{Y}_i = \mathcal{F}_2(\Phi_d; \mathcal{W}_2) \tag{8}$$

The selection of $\rho(x_i)$ in Equation (5) and (6), was different for each optimization step. Motivated by the work in [28], the VGG16 network was fine-tuned with the logarithmic function stated in Equation (9), where the index i is omitted for clarity in the notation, $\gamma \in \Re$ is a parameter, and ϵ a small positive constant (which avoids the evaluation of the logarithmic function at zero). This function saturates large gradients produced by the error between ground-truth and estimated data, adding robustness to the optimization. Equation (9) was applied to (5) using $\gamma = 2.0$, resulting in a function that operates over the mean squared differences between ground-truth and estimated data. In contrast, Equation (9) was applied to (6) with $\gamma = 1.0$, resulting in a function that process the absolute difference of residuals. Another design choice for $\rho(x_i)$ consist of a linear function, shown in Equation (10) (where the index i is omitted), which provides better convergence during the LSTM network optimization.

$$\rho(x) = \ln\left(x^{\gamma} + \epsilon\right) \tag{9}$$

$$\rho(x) = x \tag{10}$$

5. Experiments

The proposed RCNN architecture was implemented in Python using the Tensorflow [48] framework. The experiments were carried out using multiple Graphics Processing Units (GPU), including the NVIDIA Titan X and Tesla K80. The dataset samples (including video sequences from the four cameras, tool and force data vectors) were split into the training and test sets, as detailed in Table 1.

5.1. Experiments Design

First, the VGG16 network is fine-tuned with the objective to find a feature vector representation $\phi_t^{video} \in \Re^{4096}$, for every space-time frame $U_t^{video} \in \Re^{224\times224\times3}$ (see Fig. 6). Subsequently, in the LSTM network optimization, three types of feature vectors Φ_t (processed at every time step t), were evaluated as input data:

- **Case I**. Only tool data as input: $\Phi_t = \phi_t^{tool} \in \Re^4$.
- **Case II**. Only feature vectors extracted from video sequences as input: $\Phi_t = \phi_t^{video} \in \Re^{4096}$.
- **Case III**. Both, tool data and feature vectors extracted from video sequences as input: $\Phi_t = [\phi_t^{video}, \phi_t^{tool}]' \in \Re^{4100}$.

For each aforesaid case, two loss functions were evaluated to investigate the contribution of the RMSE and GDL terms that appear in Equation (4):

- **Loss A**. Setting $\alpha = 0.75$ results in the loss $\mathcal{L} = 0.75\,\mathcal{L}_{RMSE} + 0.25\,\mathcal{L}_{GDL}$. Thus, more importance is given to the RMSE than to the GDL, due to the faster convergence of the former term compared to the latter.
- **Loss B**. Setting $\alpha = 1.0$ results in the loss $\mathcal{L} = \mathcal{L}_{RMSE}$. Therefore, only the RMSE is considered in the optimization.

Therefore, a total of six cases, following the format *case number-loss type*, were analyzed during the LSTM network optimization. These cases are referred to as I-A, I-B, II-A, II-B, III-A, and III-B.

During the RCNN optimization, the normalized ground-truth force data, in the range -0.7/+0.5, were scaled by a factor $K > 1$. This strategy avoids rendering vanishing gradients, which are unhelpful to update the neural network parameters. Specifically, with $K = 1$, the loss computed with Equation (4) becomes close to zero even at the beginning of the RCNN optimization (and consequently the gradients). Thus, during the VGG16 network fine-tuning and LSTM network optimization, the normalized ground-truth force data were scaled by a factor $K = 10$, resulting in the range -7/+5 (which includes both the training and test sets). In the inference stage, the estimated forces by the RCNN are dimensionless and, because of the scaling factor, they lie in the range -7/+3 (only the test set).

The optimization of the VGG16 and LSTM networks is detailed in Sections 5.2 and 5.3, respectively. Then, in Section 5.4, an ablation study is described, which reveals the importance of each neural network in the RCNN model. Afterward, in Section 5.5, additional experiments are detailed, whose objective is to evaluate the robustness of the proposed RCNN model. Finally, Section 5.6 explains an experiment in which a time-series model is studied in the force estimation task.

5.2. VGG16 Network Fine-tuning

The VGG16 model, with weights pre-trained on the ImageNet dataset [17], was fine-tuned with Equation (4) using the Root Mean Squared Error Propagation (RMSProp) optimizer [49], completing over 100K iterations. In particular, during this process, all the network parameters were fine-tuned, except those found in the last layer (referred to as layer O9 in Fig. 6), which were optimized from scratch. Table 2 lists the hyper-parameters used during the optimization process, which were adjusted experimentally. In particular, α was set to 0.8 due to the faster convergence

Table 1: Dataset samples used in the experiments: (a) Complete dataset including both, pushing and pulling tasks, (b) dataset describing only pushing and (c) pulling tasks.

Dataset Type	Video Sequences # Files	Video Sequences Duration[(2)]	Samples[(1)]	Percentage
(a) Complete Dataset (100% of the total data samples)				
Training	28	∼3 h 19 min	597388	77%
Test	16	∼1 h	179292	23%
Total	**44**	**∼4 h 19 min**	**776680**	**100%**
(b) Pushing Tasks (59% of the total data samples)				
Training	16	106.26 min	318776	70%
Test	12	46.48 min	139448	30%
Total	**28**	**152.74 min**	**458224**	**100%**
(c) Pulling Tasks (41% of the total data samples)				
Training	12	92.87 min	278612	87%
Test	4	13.28 min	39844	13%
Total	**16**	**106.15 min**	**318456**	**100%**

[(1)] Each sample consists of a video frame (224×224×3), a (4-dimensional) tool data vector, and a (6-dimensional) ground-truth force vector.
[(2)] Computed as $T = N/F_r$, where T is the video duration, N the total number of frames, and F_r is the frame rate (50 frames per second).

of the RMSE compared to the GDL, while ϵ was set to 1/100 for numerical stability.

The VGG16 model accuracy was evaluated with the Mean Absolute Error (MAE), shown in Equation (11), where M and N stand for the number of samples and force components, respectively. The MAE was computed in the training and test sets every 10K iterations. The model accuracy and the training loss are depicted in Fig. 7. Additionally, the evolution of the error corresponding to the j-th force component, e_j, with $j = 1, ..., N$, was calculated in the training set using Equation (12). This error is depicted in Fig. 8 on a logarithmic scale (i.e. $\ln(e_j)$) and was computed every 250 iterations.

$$MAE = \frac{1}{M}\sum_{i=1}^{M}\sum_{j=1}^{N}|Y_i^{(j)} - \widehat{Y}_i^{(j)}|, \quad MAE \in \Re \tag{11}$$

$$e_j = \sqrt{\sum_{i=1}^{M}(Y_i^{(j)} - \widehat{Y}_i^{(j)})^2}, \quad e_j \in \Re \tag{12}$$

After the VGG16 network was fine-tuned on the video dataset, visual features ϕ_t^{video} were extracted from the fully connected layer FC7 (see Fig.6), replacing the rectified linear unit by the hyperbolic tangent (Tanh) non-linearity. By applying the Tanh non-linearity, all values present in the feature vectors are squashed between ±1. This range of values is expected in the feature vectors to be processed by the LSTM network (during both training and inference stages) since the block-input of this network has the Tanh non-linearity as the activation function (as described in [21]). Each feature vector computed by the VGG16 network can be interpreted as a learned representation in the low-dimensional space ($\phi_t^{video} \in \Re^{4096}$) for each input video frame that lies in the high-dimensional space ($U_t^{video} \in \Re^{224\times224\times3}$).

Table 2: Hyperparameters used for the VGG16 model fine-tuning.

Hyperparameter	Value
Learning Rate, λ	1×10^{-5}
Batch Size, M	50 samples
Dropout (Fully-Connected Layers)	50 %
Parameter α in Equation (4)	0.8
Parameter ϵ in Equation (9)	1/100

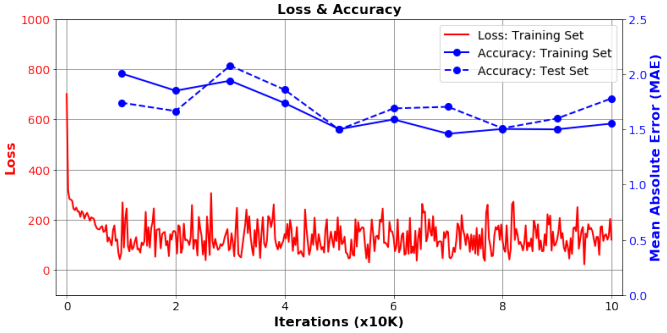


Figure 7: Computed loss (in red) and accuracy (in blue), during the fine-tuning of the VGG16 network.

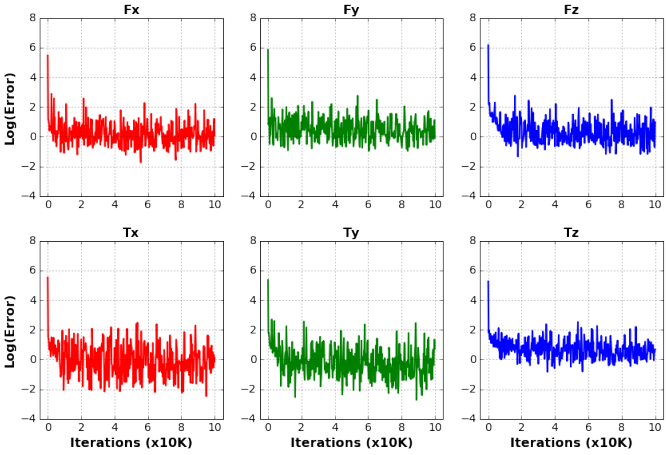


Figure 8: Logarithm of the error per force component computed (on data in the training set) during the fine-tuning process.

5.3. LSTM Network Optimization

Three models were empirically evaluated in the force estimation task: (i) The vanilla LSTM network [23] (with added peephole connections), (ii) the coupled input-forget gate variant of the LSTM network (LSTM-CIFG) [21], and (iii) the Gated Recurrent Unit (GRU) [22]. In terms of convergence and quality of prediction, the LSTM-CIFG was superior to the vanilla LSTM and GRU networks. The worst results were obtained with the GRU model. Therefore, the LSTM-CIFG network was selected to carry out the experiments and predict interaction forces between surgical instruments and tissues.

The LSTM-CIFG network was trained with the RMSProp optimizer, using the hyper-parameters listed in Table 3. For case I, this neural network was designed with only 64 cell units per layer due to the low dimensionality

Table 3: Hyperparameters used for the LSTM network optimization.

Case	**I**	**II**	**III**	**I**	**II**	**III**
Loss Function	**A**$^{(1)}$			**B**$^{(2)}$		
Number of Layers	2					
Cells per Layer	64	256	256	64	256	256
Time Steps	64					
Learning Rate, λ	0.0025					
Batch Size, M	512 samples					
Dropout L1$^{(3)}$	75%	25%	25%	75%	25%	25%
Dropout L2$^{(4)}$	75%	25%	25%	75%	25%	25%
Iterations$^{(5)}$	99.0	39.7	57.9	99.0	49.1	26.7

(1) Loss function A: $\mathcal{L} = 0.75\,\mathcal{L}_{RMSE} + 0.25\,\mathcal{L}_{GDL}$.
(2) Loss function B: $\mathcal{L} = \mathcal{L}_{RMSE}$.
(3) Dropout applied to layer 1 (L1).
(4) Dropout applied to layer 2 (L2).
(5) Total number of iterations (×1000).

of the input data ($\phi_t^{tool} \in \Re^4$), avoiding over-fitting in the training set. In contrast, the neural networks designed for cases II and III required higher capacity (i.e. more parameters) due to the complexity added by the feature vectors ($\phi_t^{video} \in \Re^{4096}$) in the input data. Therefore, these neural networks were designed with 256 cell units per layer. In all the six cases (I-A, ..., III-B), dropout was applied at the output of each layer as a method for regularization to prevent over-fitting (a higher value was set for the case I). For each case and loss function studied, the total number of iterations required to optimize the LSTM-CIFG network is shown in the last row of Table 3. The optimization was stopped after observing that the loss value reached a plateau, and there was no visible improvement in test set accuracy.

The quality of the predicted force signals with respect to the ground truth was assessed by considering two metrics, the Root Mean Square Error (RMSE) and the Pearson Correlation Coefficient (PCC).

5.4. Ablation Study

An ablation study was performed on the RCNN architecture to reveal the importance of its components. Specifically, the quality of the force vectors rendered by the VGG16 network was contrasted against that resulting from the VGG16 network serially connected with the LSTM-CIFG network. The quality of these vectors was measured with the mean absolute error, defined in Equation (11), using samples from the test set.

5.5. Robustness of the RCNN Model

Two experiments, described below, were carried out to evaluate the robustness of the RCNN model.

In the first experiment, the robustness of the RCNN model against noise, z, added to normalized tool data, ϕ_t^{tool}, was evaluated. The noise, z, was sampled from a Gaussian distribution with zero mean, $\mu = 0$, and finite variance, σ^2. Thus, $z \sim \mathcal{N}(0, \sigma^2)$. This noise was designed taking into account the statistics of the tool data, specifically, its mean-squared-value (0.0972) and standard deviation (0.3114). As the noise intensity was strengthened by increasing its variance (from $\sigma^2 = 0.001^2$ to $\sigma^2 = 14^2$), the deterioration of the estimated force signal quality was measured with the PCC and RMSE metrics.

In the second experiment, the RCNN model performance was evaluated by feeding this neural network with input video sequences pre-processed in offline and real-time modes. In offline mode, the whole video sequence is available for computing and applying pre-processing operations on raw frames, namely mean frame removal and space-time transformation. In contrast, in the real-time mode, only the past frames from video sequences can be used to perform such pre-processing operations. In the context of a real-time scenario, the computation of a mean frame followed by its subtraction from a specific video sequence represents a key pre-processing operation that has an impact on the quality of the estimated force signals. Therefore, in the real-time mode, the mean frame was computed by averaging only past frames in a video sequence. On the other hand, in the offline mode, the mean frame was obtained by averaging all the frames in a video sequence (in the experiments described in Sections 5.2 and 5.3, it was assumed that all video sequences were available offline). Afterward, the quality of the force estimations that resulted from each pre-processing mode was compared. Two samples of video sequences (from the test set) were used in this experiment, each one related to pushing and pulling tasks. This analysis reveals that the RCNN model is suitable for the task of force estimation in real-time. However, there is a small degradation of the quality of the estimated force signals with respect to the offline mode. These results will be discussed in the next section.

5.6. RCNN Model vs Time Series Model

A simpler method (not based on neural networks) than the proposed RCNN was investigated in the task of force estimation. For this purpose, a Multiple-Input Multiple-Output (MIMO) Auto-Regressive Moving Average Model with eXogenous Inputs (ARMAX), commonly used in the context of time series modeling and system identification, was selected to model the complex relationship between the input tool data and the output interaction forces. The structure of this model is given in Equation (13), where $y(t)$, $u(t)$, and $e(t)$ are vectors with N_y outputs, N_u inputs, and N_e disturbances (at the time instant t), respectively. The polynomial matrices, $A(q^{-1}) \in \Re^{N_y \times N_y}$, $B(q^{-1}) \in \Re^{N_y \times N_u}$ and $C(q^{-1}) \in \Re^{N_y}$, are defined as a function the shift operator, q. The matrix $A(q^{-1})$ of order r and parameters a_1, a_2, ..., a_r, is shown in Equation (14), while $B(q^{-1})$ of order s and parameters b_0, b_1, ..., b_s, in Equation (15). In this study $C(q^{-1}) = I$, being I the identity matrix, as shown in Equation (16). The disturbance vector, $e(t)$, represents a source of white-noise with variance 1.0.

$$A(q^{-1})\ y(t) = B(q^{-1})\ u(t) + C(q^{-1})\ e(t) \tag{13}$$

$$A(q^{-1}) = 1 + a_1 q^{-1} + a_2 q^{-2} + ... + a_r q^{-r} \tag{14}$$

$$B(q^{-1}) = b_0 + b_1 q^{-1} + b_2 q^{-2} ... + b_s q^{-s} \quad (15)$$

$$C(q^{-1}) = I \quad (16)$$

Equation (13) was implemented in SCILAB [50], and its parameters (a_1, a_2 ..., a_r, b_0, b_1,..., b_s) were estimated with the method described in [51]. During the optimization stage, the ground-truth force ($Y \in \Re^6$) and tool data ($\phi_t^{tool} \in \Re^4$) were processed at the current and previous time steps, by setting $r > 0$ and $s > 0$ in Equations (14) and (15), respectively. Specifically, the ARMAX model performed better with $r = 0$, which discards the autoregressive component of the output $y(t)$, and $s = 255$ (which represents an optimal trade-off between time complexity vs accuracy), enforcing the processing of input samples $u(t)$ at the time instants t, $t-1$, $t-2$, ..., $t-255$. Thus, in the inference stage, an output sample, $y(t)$, was estimated from the input samples $u(t)$, $u(t-1)$, $u(t-2)$, ..., $u(t-255)$, in addition to the disturbance $e(t)$, scaled by a factor of 1×10^{-4}. In terms of parameters, the complexity of the ARMAX model is lower with respect to the RCNN architecture. However, it is expected the ARMAX model to describe, up to some extent, the relationship between low dimensional data, such as the tool and force data.

6. Results & Discussion

The results and discussion of the experiments are presented in five sections. First, Section 6.1 describes the results of the LSTM-CIFG network optimization (which outputs the estimated interaction force, $\widehat{Y}_t$, given as input the feature vectors, Φ_t) and discusses the six cases studied (I-A, ..., III-B). Then, Section 6.2 presents the results of the ablation study detailed in Section 5.4. Subsequently, Section 6.3 reports the results from the experiments related to the robustness of the RCNN model in the conditions described in Section 5.5. Afterward, Section 6.4 contrasts the force estimation quality of the RCNN model against the ARMAX model. Finally, Section 6.5 discusses the key ideas to improve the RCNN model in the context of real applications. All the results shown in Tables 4, 6, and 7, and Figs. 9-12, were computed using the normalized signals provided by the RCNN, which are dimensionless and in the range -7/+3. On the other hand, Table 5 shows the force estimation quality, measured with the RMSE, in physical units.

6.1. Estimated Force Signals

After the LSTM-CIFG network optimization was completed, the quality of the estimated force signals (in the test set) was measured with the RMSE and PCC metrics. These metrics are shown in Fig. 9 for each surgical task (pushing and pulling), case (I, II and III) and loss function (loss A and B). From this illustration, case III-A stands out as the best model (solid line in red color), since it has higher PCC values and lower RMSE values with respect to the other cases. On the other hand, the metrics for case III-B (dotted line in dark red color) fall behind those attributed to case III-A in a pushing task (left column), while for pulling tasks (right column) they are close in proximity. For cases II-A (solid line in green color) and II-B (dotted line in dark green color), the PCC and RMSE values are slightly behind the accuracy reported for case III-A. Therefore, the second best model could be either, case II-A or II-B, since their values are very close to each other. Finally, cases I-A (solid line in blue color) and I-B (dotted line in dark blue color), represent the worst models. This conclusion is also justified in Table 4, which presents, for every studied case (I-A, ..., III-B), the maximum, minimum and mean values computed from the metrics (corresponding to the six force components) displayed in Fig. 9 (the best values are highlighted in bold). The results presented in Fig. 9 and Table 4 suggest that the RCNN performs best when it is optimized with a loss function explicitly designed to model smooth and sharp details found in time-varying signals. In this work, the RMSE and GDL were used to promote such behavior, allowing the modeling of smooth and sharp (i.e. signal peaks) details attributed to force/torque signals. Nonetheless, other distance functions could potentially be applied for the same purpose. Moreover, these results show that it is important to provide the RCNN with both video sequences and tool data during the training and inference stages.

The force estimation quality (from the test dataset) corresponding to case III-A (with the highest accuracy) is described in Fig. 10 and Table 5. The neural network output vs target plot and the PCC are shown in Fig. 10, while the RMSE in force and torque units is reported in Table 5.

Table 4: Maximum, minimum, and mean values of the Pearson Correlation Coefficient (PCC) and Root Mean Squared Error (RMSE) metrics (shown in Fig. 9) computed for every studied case (I-A, I-B, ..., III-B), across the six force components.

Case	**Pushing Task**			**Pulling Task**		
	Max	**Min**	**Mean**	**Max**	**Min**	**Mean**
	PCC (Values closer to 1.0 are better)					
I-A	0.3800	-0.1351	0.0450	0.2110	-0.1732	0.0636
I-B	0.3655	0.0406	0.1263	0.4901	-0.0241	0.2232
II-A	0.8877	0.2474	0.5175	0.7002	**0.5492**	0.6100
II-B	0.8869	0.2405	0.5097	0.7086	0.5342	0.6024
III-A	**0.8957**	**0.2674**	**0.5466**	**0.7164**	0.5252	**0.6280**
III-B	0.8469	0.1841	0.4016	0.6860	0.5367	0.6141
	RMSE (Values closer to 0.0 are better)					
I-A	1.1997	0.3502	0.6407	0.8517	0.4329	0.6509
I-B	1.3149	0.2785	0.5672	0.8278	0.4349	0.6313
II-A	**0.4531**	0.1732	0.3137	0.7043	0.3321	0.5195
II-B	**0.4531**	0.1726	0.3098	0.6962	0.3419	0.5161
III-A	0.4567	**0.1598**	**0.3038**	0.6778	**0.3199**	**0.5041**
III-B	0.6592	0.2596	0.3967	**0.6756**	0.3320	0.5168

Table 5: Case III-A: Root Mean Squared Error (RMSE), where the force and torque units are expressed in Newtons (N) and Newtons per meter (Nm), respectively.

Task	$\boldsymbol{F_x}$	$\boldsymbol{F_y}$	$\boldsymbol{F_z}$	$\boldsymbol{T_x}$	$\boldsymbol{T_y}$	$\boldsymbol{T_z}$
Pushing	0.0615	0.0446	0.5536	0.1405	0.1810	0.0116
Pulling	0.0756	0.0914	0.4447	0.5957	0.2830	0.0191

In Figs. 9 and 10 is observed a high PCC value (0.8957) and low error present in the F_z force component related to pushing tasks. Regarding pulling tasks, the estimated force F_z has also higher PCC value (0.7164) with respect to the rest of force components. However, it falls below the PCC value reported for pushing tasks. These results suggest that interaction forces produced by pushing tasks (smooth signals) are easier to model than those generated by pulling tasks (irregular signals). A possible explanation of these results can be deduced from the video frames computed in the space-time domain, depicted in Fig. 4.Thus, when dealing with pushing tasks, tool-tissue interactions seem to be regular and independent of the organs' geometry. For instance, the point of interaction is defined by a small contact area with an oval shape (Fig. 4a). In contrast, those interactions resulting from pulling tasks are more irregular and highly dependent on the organs' geometry (Fig. 4b). The slightly imbalance in the dataset samples that represent each surgical task, may be a small contributing factor for this result (59% and 41% of the dataset samples correspond pushing and pulling tasks, respectively, as shown in Table 1).

The results of Table 5 show the potential of the proposed RCNN architecture, upon which new models can be devised. For real operational purposes, the RMSE for forces is reported to fall below 0.1 N in both vision-based [38] and prototyped sensors [52].

A sample of estimated forces (from the test dataset) between the surgical instrument and the tissue (normalized in the range -7/+3), related to case III-A is shown in Fig. 11a and Fig. 11b for pushing and pulling tasks, respectively. Fig. 11a shows that the amplitude of most interaction forces (estimated for pushing tasks) are close to zero, with the exception of the F_z force component. The reason is that the forces are mainly applied along the surgical instrument shaft which is aligned with the z axis of the force sensor. It is also observed that the estimated shape of F_z is fully retrieved, although its amplitude differs in some locations from the ground-truth signal. By contrast, in Fig. 11b, the force and torque components (estimated for pulling tasks) are non-zero, because of the reaction forces applied to the surgical instrument when it is grasping a tissue. Nonetheless, these signals are more difficult to learn in both amplitude and shape.

6.2. Ablation Study

The ablation study reveals that the force vectors estimated by the RCNN, corresponding to case III-A, have higher quality than those estimated by the VGG16 network alone. In particular, the mean absolute error (computed with Equation (11), using force data samples normalized in the range -7/+3), is 7.5× times lower for the RCNN (~ 0.237) than for the VGG16 network (~ 1.780). This result suggests that the LSTM-CIFG network is an essential component of the force estimation model, and shows the importance of modeling the structure of data over the temporal dimension.

6.3. Robustness of the RCNN Model

The results of the robustness of the RCNN model against noise, $z \sim \mathcal{N}(0, \sigma^2)$, added to normalized tool data, ϕ_t^{tool}, are shown in Fig. 12. In this illustration, it can be observed that the PCC and RMSE metrics are deteriorated by a small margin as the noise intensity is strengthened (by increasing σ from 0.001 to 14). Nonetheless, this effect is more noticeable in the metrics related to pushing tasks than those of pulling. These results suggest that the RCNN model is able to cope with tool data corrupted with Gaussian noise, with zero mean and finite variance. Furthermore, they reveal that the estimation of interaction forces heavily relies on the input video sequences.

The comparison of the RCNN performance by preprocessing video sequences in offline and real-time modes is summarized in Table 6. The metrics reported in this table correspond to a pair of video sequences in the test set, and each video sequence is related to pushing and pulling tasks. These metrics reveal a slight deterioration of RCNN model performance in real-time mode (referred to as RT) with respect to the offline mode (referred to as O). The percentage error (calculated with respect to the offline mode and indicated with δ_p) emphasizes this result, showing a small performance gap between the two modes. Contrary to the intuition, few metrics seem to favor the RCNN model operating in real-time mode, however, they do not represent the most important force components for each surgical task. A possible explanation for this result could be related to the amount of noise present in video sequences after the pre-processing stage. That is, video sequences pre-processed in real-time mode are noisier than those pre-processed in offline mode. Such noise could be beneficial for the RCNN while operating in real-time mode.

6.4. RCNN Model vs ARMAX Model

The ARMAX model and two variants of the RCNN, referred to as cases III-A (where both video sequences and tool data are processed) and I-B (in which only tool data is processed), are contrasted in Table 7. Specifically, this table shows the PCC and RMSE computed from the estimated force signals (data in the test set), for each model and surgical task (pushing and pulling). The PCC and RMSE values presented in this table reveal that the RCNN model corresponding to case III-A, is a better choice than the ARMAX model in the task of force estimation. On the other hand, the ARMAX model outperforms the RCNN corresponding to case I-B. This result highlights the importance of processing past information. That is, although the ARMAX model (with 6144 parameters) has fewer parameters than the RCNN defined by case I-B (with 38662 parameters), the former model processes 256 input samples (at time steps t, $t-1$, ..., $t-255$) to render a single force estimate, while the later only has access to 64 input samples (at time steps t, $t-1$, ..., $t-63$). Moreover, the experimental findings suggest that the information encoded in the tool data is not enough to render accurate force

Table 6: Comparison of the RCNN model performance in offline (O) and real-time (RT) modes, using Pearson Correlation Coefficient (PCC) and Root Mean Squared Error (RMSE). The percentage error, δ_p, shows the performance gap between the two modes.

Mode	F_x	F_y	F_z	T_x	T_y	T_z
Pushing Task						
	PCC					
O	0.5816	**0.4869**	**0.9286**	**0.5860**	**0.8643**	0.2432
RT	**0.5873**	0.4546	0.8794	0.5480	0.8205	**0.2611**
δ_p	0.99%	6.64%	5.29%	6.49%	5.06%	7.34%
	RMSE					
O	**0.1797**	**0.2182**	**0.4528**	**0.1103**	**0.1113**	0.3874
RT	0.1817	0.2209	0.5918	0.1164	0.1260	**0.3864**
δ_p	1.14%	1.22%	30.69%	5.56%	13.27%	0.26%
Pulling Task						
	PCC					
O	**0.7134**	0.6635	**0.7070**	**0.6700**	0.7214	**0.5935**
RT	0.6838	**0.6845**	0.6547	0.6654	**0.7238**	0.5637
δ_p	4.14%	3.16%	7.40%	0.69%	0.34%	5.03%
	RMSE					
O	**0.3079**	0.5915	**0.3737**	0.6435	**0.3423**	**0.6555**
RT	0.3217	**0.5814**	0.4009	**0.6431**	0.3489	0.6691
δ_p	4.48%	1.70%	7.30%	0.07%	1.92%	2.07%

O (RT): Metric computed in offline (real-time) mode with respect to the ground-truth force data.
δ_p: Percentage error, computed by taking values in offline mode as a reference, i.e. $\delta_p = (|RT - O|/O) \times 100\%$.

Table 7: Comparison of the RCNN (cases III-A and I-B) vs ARMAX model using the Pearson Correlation Coefficient (PCC) and Root Mean Squared Error (RMSE) as performance metrics.

Model	F_x	F_y	F_z	T_x	T_y	T_z
Pushing Task						
	PCC					
RCNN$^{(1)}$	**0.5864**	**0.4537**	**0.8957**	**0.4246**	**0.6520**	**0.2674**
RCNN$^{(2)}$	0.1169	0.0479	0.3655	0.0551	0.0406	0.1317
ARMAX	0.2705	0.2254	0.6499	0.0909	0.2468	0.1974
	RMSE					
RCNN$^{(1)}$	**0.2603**	**0.3025**	**0.4567**	**0.1598**	**0.2366**	**0.4072**
RCNN$^{(2)}$	0.4181	0.5407	1.3149	0.2785	0.3520	0.4992
ARMAX	0.3040	0.3361	0.7462	0.2249	0.2900	0.4288
Pulling Task						
	PCC					
RCNN$^{(1)}$	**0.6917**	**0.5993**	0.7164	**0.5824**	**0.6530**	**0.5252**
RCNN$^{(2)}$	0.2720	0.1646	0.4901	-0.0241	0.1431	0.2935
ARMAX	0.5008	0.1639	**0.7616**	0.0486	0.1692	0.1268
	RMSE					
RCNN$^{(1)}$	**0.3199**	**0.6200**	**0.3669**	**0.6778**	**0.3698**	**0.6703**
RCNN$^{(2)}$	0.4349	0.7506	0.5060	0.8278	0.4928	0.7758
ARMAX	0.3980	0.7493	0.3784	0.8164	0.4789	0.7646

$^{(1)}$ Case III-A: Video sequences and tool data are processed as input.
$^{(2)}$ Case I-B: Only tool data is processed as input.

estimates, but it should be processed together with video sequences. In this context, the RCNN defined by case III-A, stands out since it can process (high-dimensional) data with both spatial and temporal components. On the other hand, the ARMAX model is limited to process (low-dimensional) temporal data.

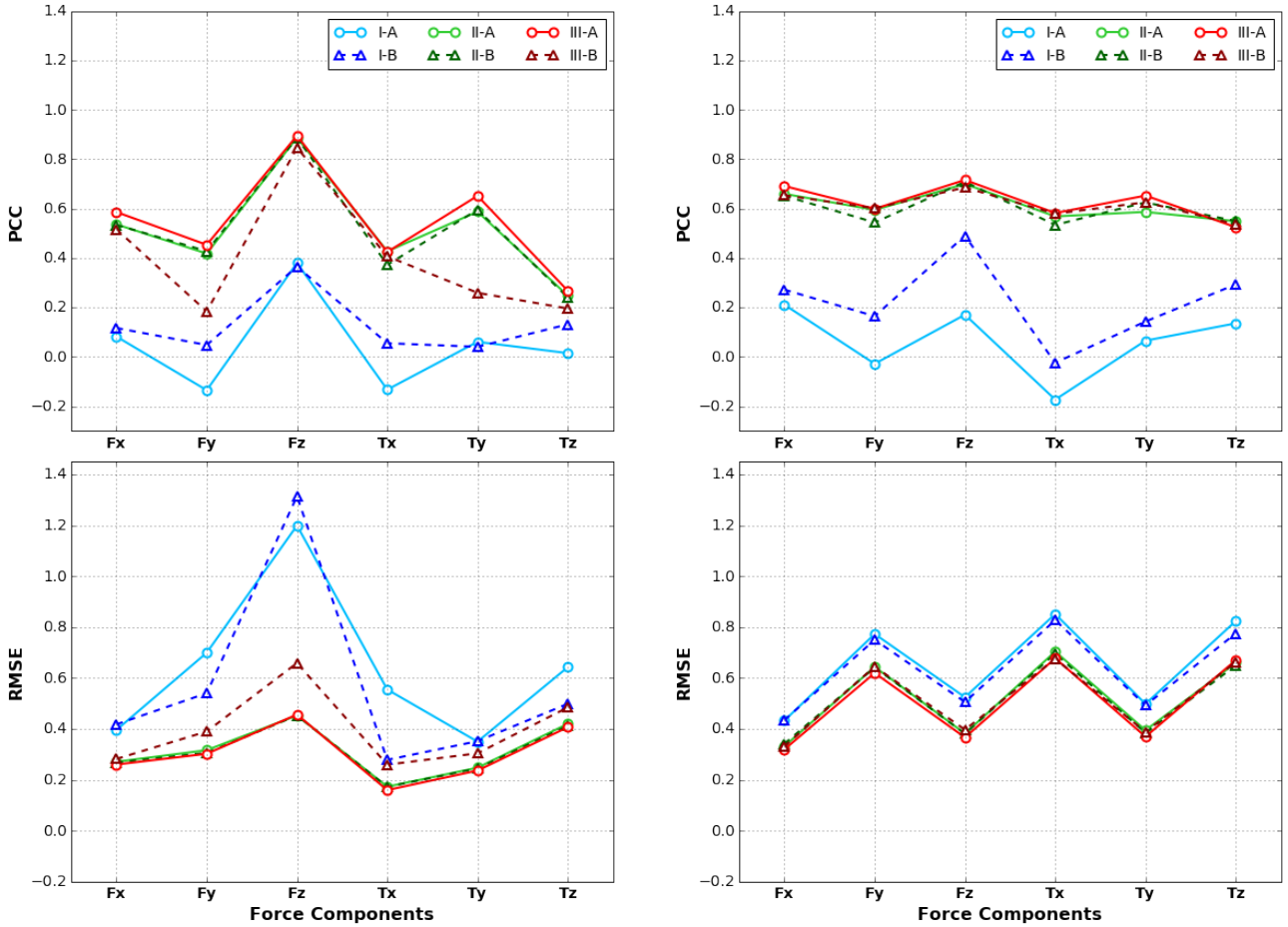
6.5. Requirements for Real Applications

For practical applications, there are four key features of the RCNN model that should be improved. First, the error reported in Table 5, can be reduced (to meet the design requirement of 0.1 N for forces) by taking into account the processing of depth information. This information can help to improve the quality in the force estimates, similarly in that the addition of tool data (i.e. the tool-tip trajectory and its grasping status) helped to render force estimates with better quality than processing only video sequences. For this purpose, a monocular depth estimation technique, such as [53], can be used. Second, techniques for preprocessing of video sequences were explored as a first approach to highlight motion due to tool-tissue interactions and ease the learning process of the neural network model. However, an attention model, such as the one described in [54], represents a suitable approach to automatically learn those image regions that are relevant to the task of interest (force estimation). Third, to circumvent the limitation of processing low-resolution images (i.e. 224×224 pixels) due to hardware constraints (i.e. the GPU memory), images with higher resolutions (i.e. 1024×1024 pixels) can be processed in patches (i.e. 256×256 pixels), as suggested in [55], in the task of image translation. Finally, the RCNN, consisting of the VGG16 network connected in series with the LSTM-CIFG network, results in a model with many parameters, which is slow during both training and inference stages. For real-time scenarios, a compact model is needed, capable of rendering force estimates without loosing quality. To this end, techniques for compressing and accelerating deep neural networks can be useful. For instance, parameter pruning and sharing, low-rank factorization, transferred/compact convolutional filters, and knowledge distillation [56].

7. Conclusions & Future Work

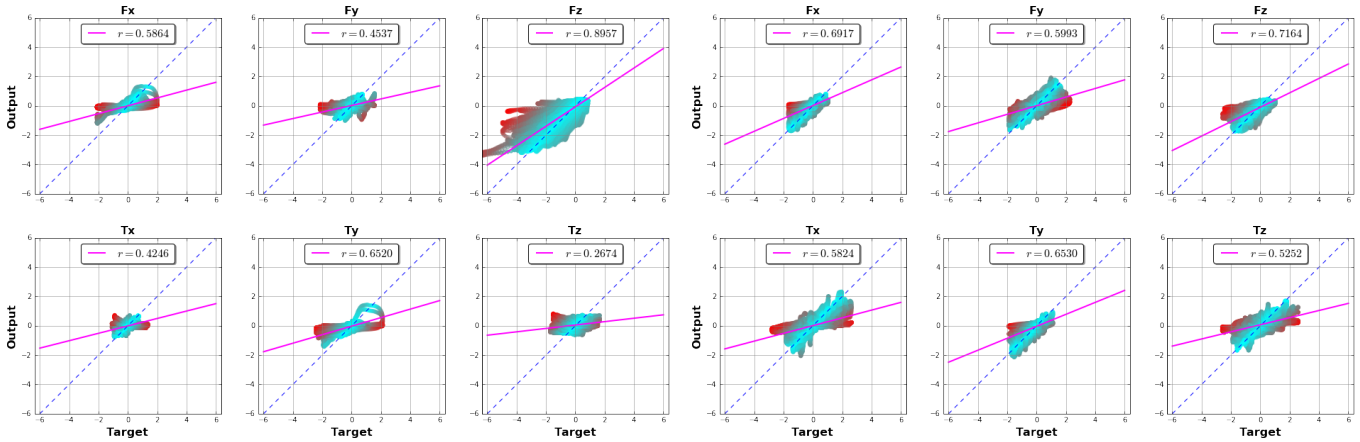
A Recurrent Convolutional Neural Network (RCNN) for Vision-Based Force Sensing (VBFS) in robotic surgery has been developed. The proposed neural network was designed to estimate forces from monocular video sequences, as opposed to the majority of reported works, which rely on stereo vision. For this purpose, a pre-trained CNN was used to learn a compact feature vector representation for each frame in a video sequence (ϕ_t^{video}), which encodes complex phenomena such as deformation of soft-tissues and specular reflections. This representation together with the tool data (ϕ_t^{tool}), defined a new feature vector space (i.e. by concatenating ϕ_t^{video} and ϕ_t^{tool}), increasing the quality in the force estimates. To enforce a temporal constraint, the feature vector space was modeled by an LSTM network. The proposed RCNN model represents an alternative to existing approaches and has the potential to achieve better results in the future.

From this research work, several experimental findings can be highlighted. First, the force estimation task is achieved better when the CNN and LSTM networks are optimized with a loss function that takes into account the Root Mean Squared Error (RMSE) and Gradient Difference Loss (GDL). The intuition behind this loss function design is that continuous and time-varying signals can be interpreted as composed of smooth and sharp details. Therefore, the RMSE addresses the modeling of smooth



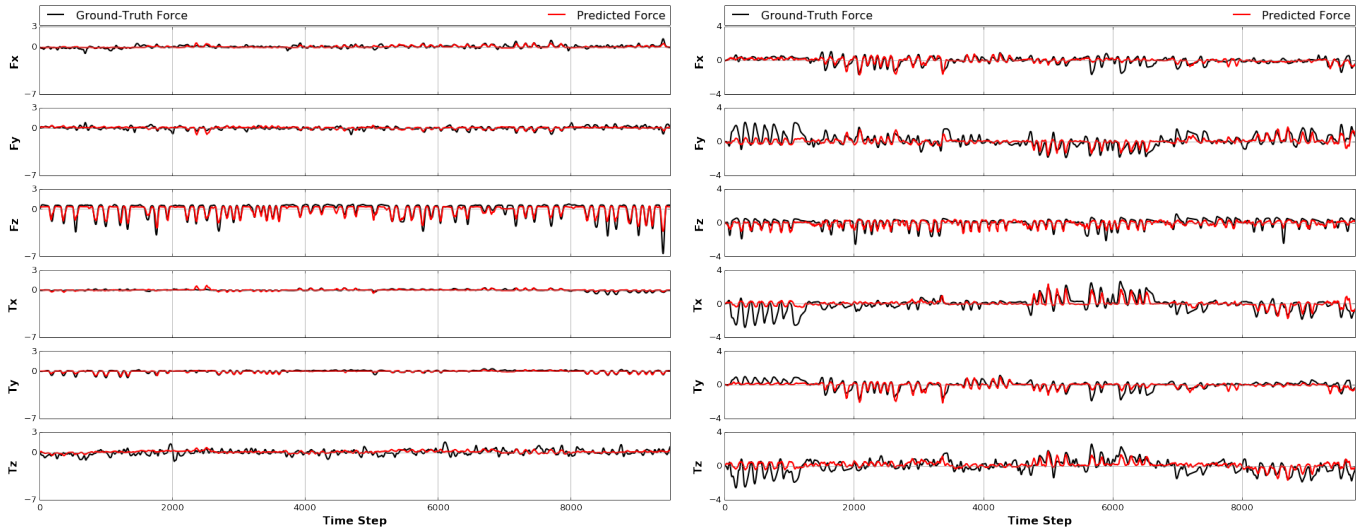
(a) Pushing Tasks (b) Pulling Tasks

Figure 9: Force estimation quality measured with the Root Mean Squared Error (RMSE) and Pearson Correlation Coefficient (PCC) for each surgical task, pushing (left column) and pulling (right column) tissue. The six cases studied (I-A, I-B, II-A, II-B, III-A, and III-B) are contrasted in these plots. For the PCC, values closer to 1.0 are better, while for the RMSE values closer 0.0 are desirable. In this illustration, case III-A (solid line in red color) stands out at the best model.



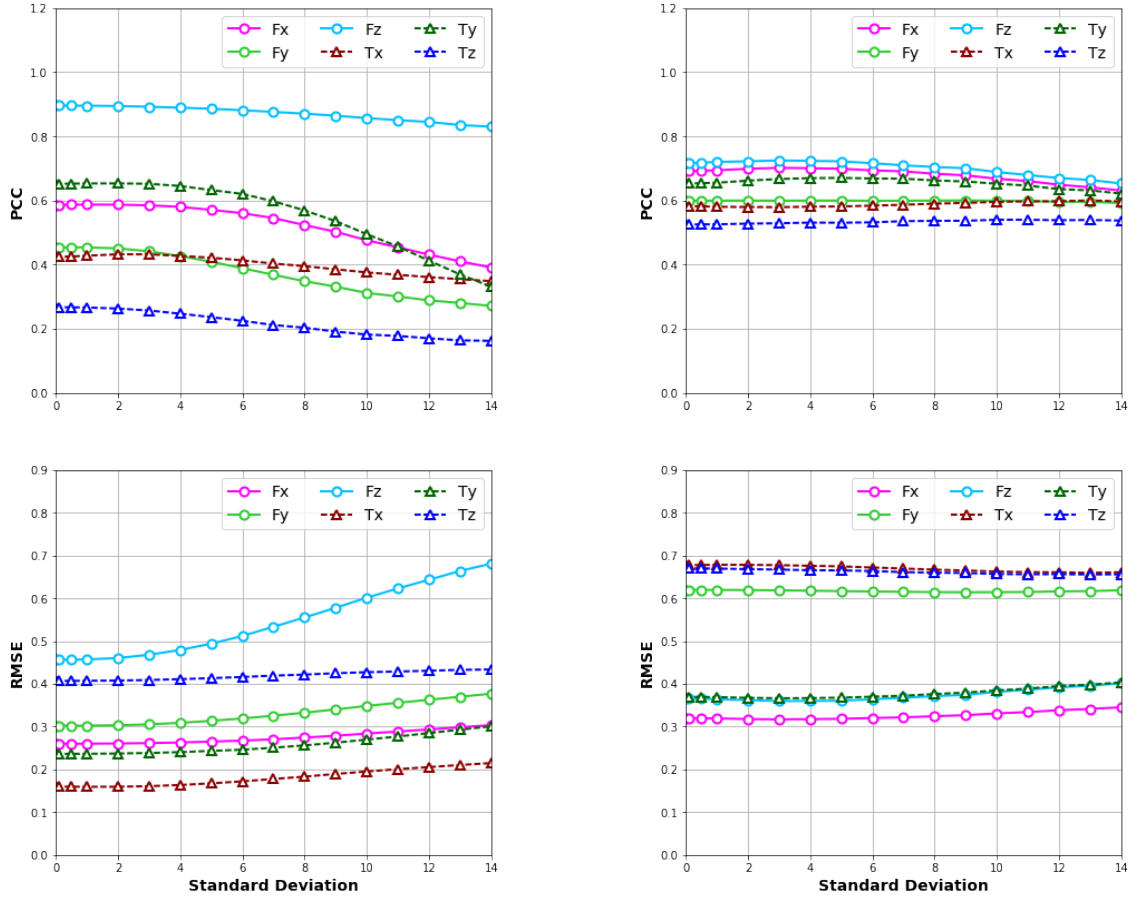
(a) Pushing Task (b) Pulling Task

Figure 10: Case III-A: Neural network output vs target plot (for all data in the test set) related to pushing (left column) and pulling tasks (right column). The Pearson Correlation Coefficient (PCC) is shown for each force component as r. The best line that fits the data is shown in magenta color. A perfect fitting to the data is represented by the dotted line in dark blue color. Data points with low and high error are plotted in blue and red colors, respectively.



(a) Pushing Task (b) Pulling Task

Figure 11: Case III-A: Sample of estimated interaction forces between tool and tissue (normalized in the range -7/+3) for pushing (left column) and pulling tasks (right column).



(a) Pushing Tasks (b) Pulling Tasks

Figure 12: Case III-A: Deterioration of the RCNN model as noise, $z \sim \mathcal{N}(0, \sigma^2)$, is added to normalized tool data, ϕ_t^{tool}, with increased strength (by varying the standard deviation, σ). The Pearson Correlation Coefficient (PCC) and Root Mean Squared Error (RMSE) metrics (per force component) related to pushing and pulling tasks, are shown on the left and right columns, respectively.

information found in force/torque signals (i.e. sine wave-like signals), while the GDL promotes the modeling of sharp details attributed to these signals (i.e. signal peaks). However, other alternatives to the GDL may result in better outcomes. For instance, the adversarial loss, which is derived from the Generative Adversarial Network (GAN) framework [57], has proven useful in the modeling of high-frequency components found in images. This type of loss can be adapted to the modeling of sharp details found in force/torque signals. Second, both video sequences and tool data, provide important cues for the estimation of forces than using either source of information alone. Third, this study shows that interaction forces resulting from pushing tasks (characterized by smooth signals) are easier to model and estimate than those produced by pulling tasks (characterized by irregular signals). Fourth, the experiment related to the robustness of the RCNN against Gaussian noise added to the tool data suggests that the RCNN model is able to cope with this perturbation. Furthermore, this experiment shows that the RCNN relies heavily on video sequences to estimate interaction forces. Fifth, regarding the pre-processing of video sequences in real-time, this experiment shows that the RCNN model performance is slightly degraded with respect to that relying on video sequences pre-processed offline. Finally, the ARMAX model is unable to render accurate force estimates by processing only tool data. The information encoded in video sequences is essential in the task of force estimation. In this context, the RCNN stands out, since it can process both video sequences and tool data, outperforming the accuracy of the ARMAX model.

The RCNN model presented in this work addresses a special case of real surgical scenarios. The camera and organs are static while the surgical instrument is in motion. The proposed RCNN model has been evaluated only in static scenarios, using a dataset enriched with video sequences recorded from different viewpoints. This allows the neural network to learn the relation between tool-tissue interactions and force under a variety of perspectives. A real scenario is usually more dynamic, with the camera moving automatically or at surgeon's will. Moreover, the organs may be affected by physiological motion due to breathing and heart beating cycles. Another important remark is that the RCNN model has been trained and validated in a single dataset. Thus, a single validation dataset was used, i.e. the test set, which only gives an estimate of the risk [58]. A future version of this work will provide a validation of the proposal in multiple and diverse datasets. Moreover, cross-validation will be included as an estimate of the model performance.

As future work, several research directions can be explored. Some of them have already been discussed in Section 6.5. First, for real operational purposes, the force estimation quality, shown in Table 5, could be improved by taking into account depth information (i.e. using a technique, such as [53]). Second, a model designed in a semi-supervised learning setting using an Auto-Encoder network and GANs, represents a potential approach to find a suitable feature vector representation from video sequences when few data are available. Third, incorporating an attention model [54], would allow automatically select those regions in video sequences that contribute to the learning process (i.e. where tool-tissue interactions are present), avoiding the need of applying pre-processing operations (i.e. mean frame removal and space-time transformation). Moreover, this attention mechanism would allow the extension of the neural network model to the estimation of forces related to more complex surgical tasks than pushing and pulling (i.e. suturing or knot-tying) and its application to dynamic scenarios (i.e. by processing motion due to uniquely tool-tissue interactions, while suppressing the motion caused by the camera and organs). Fourth, images with arbitrary resolutions can be processed in patches, as suggested in [55]. This technique will be helpful in the processing of high-resolution images, under specific hardware constraints, such as the GPU memory. Fifth, the impact of the receptive field of the CNN, used as a feature vector extractor, will be studied. Specifically, a CNN with small kernels (i.e. 3×3), represents a suitable design choice for estimating forces caused by pushing actions (due to the localized area of the tool contacts in the image), whereas a CNN with large kernels (i.e. 5×5), would be a better design choice in the modeling of forces caused by pulling actions (because the tool-tissue interactions appear distributed across the image). Sixth, techniques for compressing and accelerating deep neural networks should be investigated. They will help in designing a compact neural network model suitable for real-time scenarios. Finally, a better understanding of the RCNN model, e.g., by interpretation of its predictions [59, 60], will certainly help in designing more efficient RCNN architectures in the future.

Acknowledgment

The first author of this work acknowledges the Mexican National Council for Science and Technology (CONACYT) and the Mexican Secretariat of Public Education (SEP) for their support in doctoral studies. This work was supported by the German Ministry for Education and Research as Berlin Big Data Center (01IS14013A) and Berlin Center for Machine Learning (01IS18037I). This work was also supported by the Ministerio de Economía y Competitividad and the Fondo Europeo de Desarrollo Regional, ref. DPI2015-70415-C2-1-R (MINECO/FEDER).

References

[1] J. H. Palep, Robotic assisted minimally invasive surgery, Journal of Minimal Access Surgery 5 (1) (2009) 1–7.

[2] P. Gomes, Surgical robotics: Reviewing the past, analysing the present, imagining the future, Robotics and Computer-Integrated Manufacturing 27 (2) (2011) 261–266.

[3] A. Marbán, A. Casals, J. Fernández, J. Amat, Haptic feedback in surgical robotics: Still a challenge, in: ROBOT2013: First Iberian Robotics Conference: Advances in Robotics, Vol. 1, Springer International Publishing, 2014, pp. 245–253.

[4] B. Bayle, M. Joinié-Maurin, L. Barbé, J. Gangloff, M. de Mathelin, Robot interaction control in medicine and surgery: Original results and open problems, in: Computational Surgery and Dual Training: Computing, Robotics and Imaging, Springer, New York, 2014, pp. 169–191.

[5] A. M. Okamura, L. N. Verner, T. Yamamoto, J. C. Gwilliam, P. G. Griffiths, Force feedback and sensory substitution for robot-assisted surgery, in: Surgical Robotics: Systems Applications and Visions, Springer US, 2011, pp. 419–448.

[6] A. M. Okamura, Haptic feedback in robot-assisted minimally invasive surgery, Current Opinion in Urology 19 (1) (2009) 102–107.

[7] D. H. Lee, U. Kim, T. Gulrez, W. J. Yoon, B. Hannaford, H. R. Choi, A laparoscopic grasping tool with force sensing capability, IEEE/ASME Transactions on Mechatronics 21 (1) (2016) 130–141.

[8] B. Hannaford, J. Rosen, D. W. Friedman, H. King, P. Roan, L. Cheng, D. Glozman, J. Ma, S. N. Kosari, L. White, Raven-ii: An open platform for surgical robotics research, IEEE Transactions on Biomedical Engineering 60 (4) (2013) 954–959.

[9] S. M. Yoon, M.-C. Lee, C. Y. Kim, Sliding perturbation observer based reaction force estimation method in surgical robot instrument, in: Intelligent Robotics and Applications: 6th International Conference, ICIRA 2013, Proceedings, Part I, Springer, Berlin Heidelberg, 2013, pp. 227–236.

[10] Y. Li, M. Miyasaka, M. Haghighipanah, L. Cheng, B. Hannaford, Dynamic modeling of cable driven elongated surgical instruments for sensorless grip force estimation, in: IEEE International Conference on Robotics and Automation (ICRA), 2016, pp. 4128–4134.

[11] A. Krizhevsky, I. Sutskever, G. E. Hinton, Imagenet classification with deep convolutional neural networks, in: Advances in Neural Information Processing Systems 25, 2012, pp. 1097–1105.

[12] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, in: International Conference on Learning Representations, 2015, pp. 1–14.

[13] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 770–778.

[14] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich, Going deeper with convolutions, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2015, pp. 1–9.

[15] J. Yosinski, J. Clune, Y. Bengio, H. Lipson, How transferable are features in deep neural networks?, in: Advances in neural information processing systems, 2014, pp. 3320–3328.

[16] A. Esteva, B. Kuprel, R. A. Novoa, J. Ko, S. M. Swetter, H. M. Blau, S. Thrun, Dermatologist-level classification of skin cancer with deep neural networks, Nature 542 (7639) (2017) 115.

[17] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, L. Fei-Fei, ImageNet Large Scale Visual Recognition Challenge, International Journal of Computer Vision (IJCV) 115 (3) (2015) 211–252.

[18] X. Long, C. Gan, G. de Melo, J. Wu, X. Liu, S. Wen, Attention clusters: Purely attention based local feature integration for video classification, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 7834–7843.

[19] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural Computation 9 (8) (1997) 1735–1780.

[20] A. Graves, N. Jaitly, Towards end-to-end speech recognition with recurrent neural networks, in: Proceedings of the 31st International Conference on Machine Learning (ICML), 2014, pp. 1764–1772.

[21] K. Greff, R. K. Srivastava, J. Koutnk, B. R. Steunebrink, J. Schmidhuber, LSTM: A search space odyssey, IEEE Transactions on Neural Networks and Learning Systems 28 (10) (2017) 2222–2232.

[22] J. Chung, C. Gulcehre, K. Cho, Y. Bengio, Empirical evaluation of gated recurrent neural networks on sequence modeling, in: NIPS 2014 Workshop on Deep Learning, 2014, pp. 1–9.

[23] A. Graves, J. Schmidhuber, Framewise phoneme classification with bidirectional LSTM networks, in: Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005., Vol. 4, 2005, pp. 2047–2052 vol. 4.

[24] S. Sharma, R. Kiros, R. Salakhutdinov, Action recognition using visual attention, in: ICLR 2016 - Workshop Track International Conference on Learning Representations, San Juan, Puerto Rico, 2016.

[25] J. Donahue, L. A. Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, T. Darrell, Long-term recurrent convolutional networks for visual recognition and description, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015, pp. 2625–2634.

[26] S. Venugopalan, M. Rohrbach, J. Donahue, R. Mooney, T. Darrell, K. Saenko, Sequence to sequence – video to text, in: Proceedings of the IEEE International Conference on Computer Vision (ICCV), 2015, pp. 4534–4542.

[27] C. Finn, I. J. Goodfellow, S. Levine, Unsupervised learning for physical interaction through video prediction, CoRR abs/1605.07157.

[28] A. Owens, P. Isola, J. McDermott, A. Torralba, E. H. Adelson, W. T. Freeman, Visually indicated sounds, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 2405–2413.

[29] J. Zhou, X. Hong, F. Su, G. Zhao, Recurrent convolutional neural network regression for continuous pain intensity estimation in video, in: 2016 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), 2016, pp. 1535–1543.

[30] X. Wang, G. Ananthasuresh, J. P. Ostrowski, Vision-based sensing of forces in elastic objects, Sensors and Actuators A: Physical 94 (3) (2001) 142–156.

[31] M. A. Greminger, B. J. Nelson, Modeling elastic objects with

neural networks for vision-based force measurement, in: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Vol. 2, 2003, pp. 1278–1283.

[32] F. Karimirad, S. Chauhan, B. Shirinzadeh, Vision-based force measurement using neural networks for biological cell microinjection, Journal of Biomechanics 47 (5) (2014) 1157–1163.

[33] J. Rosen, J. D. Brown, L. Chang, M. N. Sinanan, B. Hannaford, Generalized approach for modeling minimally invasive surgery as a stochastic process using a discrete markov model, IEEE Transactions on Biomedical Engineering 53 (3) (2006) 399–413.

[34] C. W. Kennedy, J. P. Desai, A vision-based approach for estimating contact forces: Applications to robot-assisted surgery, Applied Bionics and Biomechanics 2 (1) (2005) 53–60.

[35] W. Kim, S. Seung, H. Choi, S. Park, S. Y. Ko, J. O. Park, Image-based force estimation of deformable tissue using depth map for single-port surgical robot, in: 12th International Conference on Control, Automation and Systems (ICCAS), 2012, pp. 1716–1719.

[36] S. Giannarou, M. Ye, G. Gras, K. Leibrandt, H. J. Marcus, G.-Z. Yang, Vision-based deformation recovery for intraoperative force estimation of tool–tissue interaction for neurosurgery, International Journal of Computer Assisted Radiology and Surgery 11 (6) (2016) 929–936.

[37] A. I. Aviles, A. Marban, P. Sobrevilla, J. Fernandez, A. Casals, A recurrent neural network approach for 3d vision-based force estimation, in: 4th International Conference on Image Processing Theory, Tools and Applications (IPTA), 2014, pp. 1–6.

[38] A. I. A. Rivero, S. M. Alsaleh, J. K. Hahn, A. Casals, Towards retrieving force feedback in robotic-assisted surgery: A supervised neuro-recurrent-vision approach, IEEE Transactions on Haptics 10 (3) (2017) 431–443.

[39] E. Noohi, S. Parastegari, M. efran, Using monocular images to estimate interaction forces during minimally invasive surgery, in: IEEE/RSJ International Conference on Intelligent Robots and Systems, 2014, pp. 4297–4302.

[40] S. Bosse, D. Maniry, K.-R. Müller, T. Wiegand, W. Samek, Deep neural networks for no-reference and full-reference image quality assessment, IEEE Transactions on Image Processing 27 (1) (2018) 206–219.

[41] M. Mathieu, C. Couprie, Y. LeCun, Deep multi-scale video prediction beyond mean square error, in: International Conference on Learning Representations (ICLR), 2016.

[42] T. Pfister, K. Simonyan, J. Charles, A. Zisserman, Deep convolutional neural networks for efficient pose estimation in gesture videos, in: Proceedings of the Asian Conference on Computer Vision (ACCV), 2014, pp. 538–552.

[43] G. Picod, A. C. Jambon, D. Vinatier, P. Dubois, What can the operator actually feel when performing a laparoscopy?, Surgical Endoscopy And Other Interventional Techniques 19 (1) (2005) 95–100.

[44] Y. A. LeCun, L. Bottou, G. B. Orr, K.-R. Müller, Efficient BackProp, in: Neural networks: Tricks of the trade, Springer, 2012, pp. 9–48.

[45] Itseez, Open Source Computer Vision Library (OpenCV), `https://opencv.org/` (2018).

[46] A. Buades, B. Coll, J.-M. Morel, Non-local means denoising, Image Processing On Line 1 (2011) 208–212.

[47] V. Belagiannis, C. Rupprecht, G. Carneiro, N. Navab, Robust optimization for deep regression, in: Proceedings of the IEEE International Conference on Computer Vision (ICCV), 2015, pp. 2830–2838.

[48] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, et al., Tensorflow: Large-scale machine learning on heterogeneous distributed systems, arXiv preprint arXiv:1603.04467.

[49] T. Tieleman, G. Hinton, Lecture 6.5—RmsProp: Divide the gradient by a running average of its recent magnitude, COURSERA: Neural Networks for Machine Learning (2012).

[50] Scilab Enterprises, SCILAB: Open Source Software for Numerical Computation (Version 6.0.1), `https://www.scilab.org` (2018).

[51] P. Eykhoff, Trends and progress in system identification: IFAC Series for Graduates, Research Workers & Practising Engineers, Vol. 1, Elsevier, 2014, p. 96.

[52] U. Kim, D.-H. Lee, W. J. Yoon, B. Hannaford, H. R. Choi, Force sensor integrated surgical forceps for minimally invasive robotic surgery, IEEE Transactions on Robotics 31 (5) (2015) 1214–1224.

[53] C. Godard, O. Mac Aodha, G. J. Brostow, Unsupervised monocular depth estimation with left-right consistency, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 270–279.

[54] K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhudinov, R. Zemel, Y. Bengio, Show, attend and tell: Neural image caption generation with visual attention, in: International Conference on Machine Learning, 2015, pp. 2048–2057.

[55] P. Isola, J.-Y. Zhu, T. Zhou, A. A. Efros, Image-to-image translation with conditional adversarial networks, in: IEEE Conference on Computer Vision and Pattern Recognition, 2017.

[56] Y. Cheng, D. Wang, P. Zhou, T. Zhang, Model compression and acceleration for deep neural networks: The principles, progress, and challenges, IEEE Signal Processing Magazine 35 (1) (2018) 126–136.

[57] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, Y. Bengio, Generative adversarial nets, in: Advances in neural information processing systems, 2014, pp. 2672–2680.

[58] S. Arlot, A. Celisse, et al., A survey of cross-validation procedures for model selection, Statistics surveys 4 (2010) 40–79.

[59] S. Bach, A. Binder, G. Montavon, F. Klauschen, K.-R. Müller, W. Samek, On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation, PLOS ONE 10 (7) (2015) e0130140.

[60] G. Montavon, W. Samek, K.-R. Müller, Methods for interpreting and understanding deep neural networks, Digital Signal Processing 73 (2018) 1–15.